LA PRIMERA REVISTA DE PROGRAMACIÓN EN CASTELLANO

SOLO PROGRAMADORES

O III, NÚMERO 37

975 Ptas.

APIS PARA WINDOWS: LAS DIRECTX

Y ADEMÁS...

- Delphi 3.0
- Inteligencia Artificial
- Redes Locales
- WWW
- Visual Basic

ERRAMIENTAS DE DESARROLLO

atcom C/C++ 11

RIPTOGRAFÍA

nciones Hash. MD-5

ISIÓN ESTEREOSCÓPICA

cuarta dimensión

CONTENIDO DEL CD-ROM

- DirectX SDK 3.Øa
- NetShield y Netatalk
- Delphi Lightlib/400
- QuickReport
- Especial Linux
- Y mucho más

ALGORITMOS PARA PROGRAMAS DE AJEDREZ

TOWER

8 413042 303299

SÓLO PROGRAMADORES

Número 37
SÓLO PROGRAMADORES
es una publicación de
TOWER COMMUNICATIONS

Director Editor
Antonio M. Ferrer Abelló
aferrer@towercom.es

Directora Adjunta
Amelia Noguera
anoguera@towercom.es

Colaboradores
Enrique Díaz, Arsenio Molinero, Carlos Álvaro, Enrique de la Lastra, Francisco José Abad Cerdá, Jose Antonio Collar, Javier Sarsa, Oscar Prieto Blanco, Ernesto Schmitz, Alejandro Reyero, Emilio Postigo Rian, Chema Álvarez, J. Antonio Mendoza, Jorge Delgado Mendoza.

Jefe de Diseño
Fernando García Santamaría

Maquetación
Clara Francés

Tratamiento de Imagen
María Arce Giménez

Imagen de Portada
Fernando Núñez

Publicidad
Erika de la Riva (Madrid)
Tel.: (91) 661 42 11
Pepín Gallardo (Barcelona)
Tel.: (93) 213 42 29

Suscripciones
Isabel Bojo
Tel. (91) 661 42 11 Fax: (91) 661 43 86
suscrip@towercom.es

Servicio Técnico
TOWER LABS

Atención técnica
Oscar Casado/Javier Amado

Filmación
Megatipo

Impresión
G. Reunidas

Distribución
SGEL

Director General
Antonio M. Ferrer Abelló

Director Financiero
Francisco García Díaz de Liaño

Director de Producción
Carlos Peropadre

Directora Comercial
Carmina Ferrer
Redacción, Publicidad y Administración
C/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (MADRID)
Telf.: (91) 661 42 11 / Fax: (91) 661 43 86

La revista Sólo Programadores no tiene por qué estar de acuerdo con las opiniones escritas por sus colaboradores en los artículos firmados.
El editor prohibe expresamente la reproducción total o parcial de los contenidos de la revista y del CD-ROM que la acompaña sin su autorización escrita.

Depósito legal: M-26827-1994
ISSN: 1134-4792
PRINTED IN SPAIN
COPYRIGHT 30-11-97

EDITORIAL

Volver a las rutinas

A veces la mejor noticia es la ausencia de noticias, pero en informática no suele ser así, no estamos acostumbrados a que la empresas que se dedican a mejorar los entornos de desarrollo con los que después trabajaremos, nos abandonen y no saquen un nuevo producto o una nueva versión. Y por eso, cuando parece que no pasa nada, a Bill Gates se le ocurre comprar parte de una gran organización que lleva mucho tiempo en esto de la informática. Y seguir· introduciéndose en otra enorme parcela de este maravilloso mundo de las teclas y los chips. Esperaremos con expectación sus avances.

También el mes de septiembre suele ser un mes de promesas y de propósitos, es como un renacido año nuevo, quizás porque todos hemos sido niños y hemos visto aproximarse la llegada de las clases con algo de emoción, o por lo menos con resignación, y durante años nos hemos prometido estudiar más y llevar el curso mejor, o tal vez porque la mayoría vuelve de unas vacaciones siempre agradables. Puede que este sentimiento de tener que regresar a las obligaciones después de un periodo de descanso en el que parece que las cosas tienen menos importancia, sea el que nos hace tomarlo todo con una mezcla de resignación y alegría.

Como prometimos el mes pasado, os vamos a ayudar a afrontar este nuevo periodo con un especial, incluido dentro de los cuadernos de programación que llevamos varios números regalando con la revista, dedicado en esta ocasión a Linux, y esperamos que a todos sus adeptos les sea de mucha utilidad. Para información de sus numerosos seguidores, no podéis perderos el número especial que saldrá en los próximos días con la última distribución de Debian.

En este número os presentamos temas diferentes: damos un paseo por los algoritmos que más se utilizan en el desarrollo de programas de ajedrez, que os serán de utilidad a los osados que participen en nuestro concurso; nos metemos de lleno en la visión estereoscópica, explicando los detalles de la Cuarta Dimensión por ordenador; continuamos con la serie sobre inteligencia artificial que tantos seguidores tiene; y llegamos hasta la programación con las DirectX, entre otros temas de actualidad que nos parecen sumamente interesantes. Así esperamos que tengáis un magnífico regreso a la vida real.

Tenéis a vuestra disposición la dirección de correo electrónico de la revista: **solop@towercom.es,** donde podéis enviar vuestras sugerencias sobre los contenidos, los temas que queráis que incluyamos y los productos que os gustaría que distribuyéramos con el CD-ROM, que todos los meses regalamos.
Sois nuestros mejores colaboradores.

37 sólo PROGRAMADORES

6 Noticias y libros
NOVEDADES DEL MERCADO

Un somero análisis de las nuevas estrategias de desarrollo de Microsoft, así como la presentación de los últimos productos de las más importantes empresas del sector conforma esta sección.

11 WWW
HTML DINÁMICO CON NETSCAPE COMMUNICATOR Y JAVASCRIPT (II)

En esta sección dedicada al fascinante mundo de la Red de redes, abordaremos en esta ocasión algunas características de la nueva versión de Netscape Navigator; en este caso las capas de contenidos transparentes o translúcidos.

18 Redes Locales
FACTURACIÓN EN NOVELL NETWARE

El sistema de facturación que ofrece Novell Netware no sólo es útil para cobrar por los servicios que se ofrecen; también se puede utilizar para observar el uso que se hace de la red y repartir costes de producción entre distintos departamentos de la empresa. En este artículo veremos sus aplicaciones y la forma de instalarlo paso a paso.

25 Criptografía
FUNCIONES HASH. MD-5

Mediante las funciones Hash se resuelven problemas de criptografía que difícilmente podrían ser solucionados de cualquier otra manera. A lo largo del reportaje nos adentramos en una de las funciones más representativas de esta técnica criptográfica.

40 Técnicas Gráficas
VISIÓN ESTEREOSCÓPICA: LA CUARTA DIMENSIÓN

Aprenderemos a crear programas que sean capaces de proyectar objetos tridimensionales por medio de la visión estereoscópica.

45 Herramientas de Desarrollo
PRIMER CONTACTO CON WATCOM C/C ++ 11

La nueva versión de este compilador es un producto que contiene las herramientas y librerías necesarios para crear programas de 16 y 32 bits, destinados a ejecutarse en los sistemas operativos más populares. Con él, los programadores de C/C ++ han ganado mucho.

52
Visual Basic
BASES DE DATOS

Continuando con la serie de artículos que tratan sobre la creación de bases de datos de tipo Access, en esta ocasión veremos cómo es posible crear una nueva base de datos para el motor de bases de datos Jet Engine, definiendo ciertos objetos de acceso a datos que se correspondan con los que deseamos crear, como tablas, campos, índices o relaciones.

56
Programación de Juegos
ALGORITMOS PARA PROGRAMAS DE AJEDREZ

Coincidiendo con la celebración del Torneo de Programas de Ajedrez que desde esta revista hemos promovido, presentamos algunos de los algoritmos más utilizados en todo el mundo por los principales programadores dentro de este campo. Así mismo, en el CD-ROM que acompaña a la revista se incluyen algunos de los programas que los siguen.

63
Inteligencia Artificial
VIDA ARTIFICIAL Y ALGORITMOS GENÉTICOS

Un algoritmo genético es un mundo de vida artificial en el que la evolución sigue una determinada dirección, de manera que nos sea útil para resolver algún problema determinado. En este artículo, mediante un programa que ilustra los conceptos de los que se habla, estudiaremos a fondo este apasionante tema.

70
Delphi
DELPHI 3.0

La nueva edición de Delphi, la 3.0, constituye uno de los productos más importantes que han sido presentados recientemente. Después de pasar por la aprobación de los usuarios y de la industria, ha evolucionado y se ha confirmado como una de las mejores opciones tanto como para el programador amateur como para los equipos profesionales de aplicaciones cliente-servidor e Internet. Veremos por qué.

32 Herramientas de Programación
DIRECTX Y DIRECTX SDK 3.0a

Las librerías DirectX se convertirán en extensiones estándar de Windows; además de lo práctico que resulta su uso para los programadores por el tiempo de desarrollo ahorrado, serán incluidas con toda probabilidad en próximas versiones de Windows.

81
CONTENIDO DEL CD-ROM

Aplicaciones Linux variadas que completan el especial sobre herramientas de programación, la última entrega del especial Delphi, Site Visual Basic de Farpoint, algunos sencillos programas de ajedrez y la gama de virus de McAfee, entre otras cosas, conforman el CD-ROM que nos acompaña este número.

Noticias

Microsoft anuncia WINDOWS 98 y presenta sus nuevas estrategias para el mercado de desktop en el Microsoft Windows Platform Briefing

Seatle fue el lugar elegido para anunciar oficialmente esta nueva versión de Windows 95. Hace ya unos días de este anuncio y ahora el producto se encuentra en proceso de beta-testing. El hasta hace poco llamado Menphis, tiene desde entonces un nuevo nombre que seguro que volveremos a oír muchas veces en los meses que vienen.

Con respecto a las nuevas necesidades a las que Microsoft intenta responder desde la continua investigación que realiza en el mercado, en el evento, el Microsoft Windows Platform Briefing, se anunciaron también la aparición de nuevas tecnologías para reducir el Coste Total de Propiedad y el apoyo de cinco nuevas firmas a la tecnología Handheld PC, basada en el sistema operativo Windows CE.

En este sentido, la compañía pretende seguir avanzando en el desarrollo de los elementos que conforman uno de los sistemas operativos más utilizados del mundo, incluyendo el driver model, el sistema de archivo, la infraestructura de la red, la escalabilidad del kernel, las interfaces de programación y la interfaz de usuario.

Dentro de su estrategia para reducir los costes, Microsoft se encuentra desarrollando tecnologías avanzadas como la construida sobre la versión 5.0 del sistema operativo de red Windows NT Server. Una de estas tecnologías es IntelliMirror, que proporcionará a los usuarios de plataformas Windows de 32 bits las ventajas de la informática centralizada junto con las prestaciones propias de la informática distribuida.

La función de IntelliMirror replica los usuarios, datos, aplicaciones y estados personalizados del sistema operativo en un servidor que se basa en Windows NT 5.0, utilizando caché inteligente y sincronización centralizada.

Esto se puede traducir en que los usuarios tienen acceso a toda la información y aplicaciones independientemente que estén conectados a la red o no, asegurándose, además , que la información quedará guardada en el servidor. Esta tecnología forma parte de la iniciativa Zero Administration for Windows y desde Microsoft se seguirá investigando en ella para reducir los esfuerzos y los costes de implantar, instalar y gestionar el PC. Además, la compañía ha anunciado su intención de dar un importante impulso para rebajar el coste total de poseer y administrar aplicaciones Office, aproximadamente en la mitad.

Mediante la utilización de estas tenologías la compañía pretende que los desarrolladores puedan elegir entre lenguajes y herramientas y los usuarios finales sean capaces de conectarse fácilmente a todo tipo de información, a la vez que las empresas dispongan de un entorno flexible y fácil de administrar.

APARECE COSMO PLAYER 2.0
El cliente VRML multiplataforma de próxima generación basado en el estándar OpenGL

Cosmo Software, una compañía de Silicon Graphics, ha anunciado la aparición del cliente VRML multiplataforma de próxima generación Cosmo Player 2.0, diseñada para incluir una interfaz de usuario nueva y soportar los estándares de interfaces de programación de aplicaciones gráficas, OpenGL. Además, Cosmo Player estará disponible por primera vez también para Macintosh.

Un punto interesante para los desarrolladores con VRML es que el código nativo de Cosmo Player 2.0 es consistente en todas las plataformas (Windows, Macintoh y Unix).El producto también proporciona soporte nativo para Netscape Communicator como un plug-in de Netscape y un componente ActiveX de Explorer.

Cosmo Player es un cliente VRML 2.0 que trabaja como un plug-in para los navegadores principales, Netscape Navigator y Microsoft Internet Explorer., se basa en el estándar OpenGL, un API de programación con gráficos 2D y 3D que permite ahorrar tiempo de desarrollo y costes de la creación de aplicaciones gráficas creado por Silicon Graphics como tecnología abierta.

MICROSOFT ACEPTA EL ESTÁNDAR DE ESCRITURA ECMA PARA INTERNET
Además incluye JScript 3.0 en sus principales productos

Microsoft presentó recientemente el software de desarrollo Microsoft JScript versión 3.0, que es el primer lenguaje que sigue totalmente el estándar conocido como ECMA-262 (European Computer Manufacturers Association). Al mismo tiempo se han anunciado los primeros productos de la industria conformes con la norma ECMA, como Microsoft Internet Informacion Server 4.0, Microsoft Script Debugger y la próxima versión de Microsoft Internet Explorer 4.0

El estándar ECMA está siendo seguido en la actualidad por diversas empresas muy importantes del sector como Sun Microsystems, Borland International y Netscape Communications. Es probable que dichos fabricantes lancen en breve nuevas versiones de sus productos siguiendo el estándar ECMA.

ECMA es una asociación que está radicada en Europa para la normalización de informacíon y de sistemas de comunicación. El estándar que ha sido recientemente aprobado está basado en las reglamentaciones conjuntas de Microsoft y Netscape. JScript 3.0 es un nuevo lenguaje incluido en la beta de Microsoft Internet Explorer 4.0. Es posible bajarla de Internet de http://www.microsoft.com/jscript.

Breves

NUEVO PLUG-IN PARA ADOBE PREMIER: EL ANIMATED GIF

Uno de los principales programas de edición de sobremesa de vídeo digital no lineal para las plataformas Macintosh y Windows, Adobe Premiere 4.2, podrá utilizar el plug-in Animated GIF. Este novedoso plug-in permite a los usuarios de Adobe Premier exportar películas en forma de ficheros Graphic Interchange Format (GIF) multi-imagen para conseguir la creación de animaciones en las páginas de la Web.

De esta forma, los usuarios tendrán la posibilidad de exportar películas Adobe Premier o animaciones gráficas en forma de fichero GIF animado, que requieren menos ancho de banda y son más sencillos de realizar en la Web que las películas QuickTime o Vídeo para Windows (AVI).

SILICON GRAPHICS CONMOCIONA EL MERCADO CON LOS EFECTOS ESPECIALES DE JURASSIC PARK

Dentro de su línea en la creación de tecnología para animación y efectos especiales generados por ordenador, la compañía ha anunciado que sus estaciones de trabajo y servidores han sido utilizados por la empresa responsable de la creación de "The Lost World: Jurassic Park", la continuación de "Parque Jurásico".

La película tiene múltiples efectos especiales y un nivel mucho más elevado de interactividad en el proceso de creación. Los personajes digitales han abierto posibilidades creativas sin límite en esta película.

SYBASE continúa generando beneficios durante el segundo trimestre de 1997

Sybase no cesa de aumentar sus beneficios ni de incorporar clientes en todo el mundo. EDS o South Australian Health Comission son algunos de ellos.

La facturación de la compañía en el segundo trimestre del año fiscal 1997 ha alcanzado los 237,6 millones de dólares, significando esto unos beneficios de 4,4 millones de dólares. Es el cuarto trimestre consecutivo en que esta situación se produce, y no es para menos.

Productos tales como PowerBuilder o Sybase SQL SErver son muy bien recibidos por los clientes.

Tanto es así que nuevas versiones están siendo desarrolladas en este momento: PowerBuilder 6.0, la nueva versión de la herramienta de desarrollo para aplicaciones de negocio, incluye la capacidad de suministrar objetos PowerBuilder como componentes estándar, y la primera versión beta ha sido presentada en este momento.

Adaptive Server 11.5 Enterprise, que contituye la nueva generación de la base de datos relacional de la compañía, ha sido muy bien recibida en su segunda fase y estará disponible con toda probabilidad durante el segundo semestre de este año.

WIN CATALOG Y VOCAL CONTROL SERÁN DISTRIBUIDOS POR CASA DE SOFTWARE

Los nuevos productos de Gensoft serán distribuidos de forma exclusiva por Casa de Software. La compañía ha firmado recientemente un acuerdo con la empresa italiana de informática Gensoft para convertirse en el distribuidos en exclusiva de sus productos en España.

Gensoft es una sociedad que desarrolla, produce y distribuye software profesional y de entretenimiento en cuatro idiomas. Además trabaja en colaboración con diferentes empresas nacionales e internacionales en diversos proyectos como los siguientes:

- Cover Technology Montespertoli/Fi, donde se crean y se producen los códigos marcados como "Price Line", una marca que pertenece a Gensoft.
- Iona software, de Irlanda, por la distribución exclusiva en el mercado italiano de todos los productos Edmark, Mecc y Brighter Child.

Win Catalog es un programa creado con C++ y ODBC que se utiliza para organizar los programas que cada uno tiene en su ordenador. Trabaja tanto con Windows 95 como en 3.1, gracias al soporte Win 32s.Vocal Control permite controlar por voz todos los menús, diálogos y botones de Windows 95 y administrar las aplicaciones que hayan sido proyectadas y desarrolladas con el estándar Microsoft como Word o Excel.

TRINICOM 5100 DE SONY Nuevo sistema de videoconferencia con gráficos mejorados

Sony Grupo Productos Profesionales ha anunciado la aparición de un nuevo sistema de gama alta para videoconferencias en grupo, el llamado modelo Trinicom 5100.

Según apunta la compañía, es el primer sistema capaz de establecer video conferencias multipunto entre cuatro sedes sin tener que recurrir a un controlador MCU externo. Sony asegura que el Trinicom 5100 confirma su política de ofrecer los sitemas con mejor relación precio/prestaciones del mercado. El sistema es también el primero que admite monitores de pantalla panorámica (16:9). Además incorpora mejoras en las funciones de gráficos y pizarra compartida, que desde este modelo se pueden utilizar en la modalidad multipunto.

Estas funciones mejoradas son por ejemplo las de dibujo, digitalizaci;on e impresión, que utilizan las normas JPEG, Annex D y MMR, y pueden ser compartidas por todos los participantes de la videoconferencia consiguiendo de esta manera una mayor interactividad.

La consecuencia de soportar la pantalla panorámica con monitores 16:9 es un ahorro de espacio y de coste, ya que es posible mostrar la imagen de la sede remota y los gráficos en el mismo monitor. Si únicamente se desea ver la sede remota, la proporción 16:9 optimiza la pantalla para obtener la mejor vista posible de los interlocutores. Además se ha mejorado la facilidad de utilización del sitema, con iconos de pantalla y mandos a distancia diferentes que se adecúan a cada tipo de usuario. La actualización del software se efectúa a través de la RDSI.

CONSULTING.INTERCOM ofrecerá la familia de productos Internet/Intranet de TeamWare. Ambas empresas han firmado un acuerdo de distribución

TeamWare tiene como estrategia de productos tender hacia el entorno internet como principal plataforma y conseguir que todos los productos que no hayan sido creados específicamente para este entorno sen también web-enabled. Siguiendo dicha estrategia, TeamWare dispone de un sistema de diseño y creación de comunidades virtuales, un sistema de mensajerí a Internet y Groupware basado en protocolos nativos y estándares, que permite el acceso a todos los servicios colaborativos desde este entorno.

Productos de este tipo son TeamWarePlaza, para trabajo en grupo para PYMES, TeamWare Mail, un servidor de mensajería para Internet/Intranet, TeamWare Office, una suite de servicios colaborativos basada en módulos para entornos Internet/Intranet y otros muchos.

El acuerdo por el que TeamWARE Iberia y Consulting.Intercom para proceder a la distribución de la gama de soluciones Internet/Intranet de TeamWare, encaja dentro de la política de expansión y diversificación del canal de distribución que la compañía inició a principios de año.

Para más información consultar sobre Consulting.Intercom se pueden visitar las direcciones http://www.grupo.intercom y http://www.intercom.es.

ATTACHMATE POTENCIA SU PROGRAMA PREMIER PARTNER Dentro de su política de expansión en el mercado español

Attachmate es una compañía dedicada a proporcionar soluciones de acceso a la información corporativa, líder en software de conectividad. Como clientes tiene empresas importantes de todo el mundo, grandes corporaciones y administraciones públicas. Sus soluciones de conectividad -clientes orientados a objetos, software servidor, herramientas de desarrollo, software de gestión y hardware- son una extesnión de los sistemas de información, las aplicaciones y los datos, permitiendo a los usuarios, tanto locales como remotos, el acceso transparente a la información corporativa, su gestión y compartición.

Premier Partner es el programa corporativo de Attachmate a través del cual ofrece el uso de su tecnología y la resolución de problemas, tanto a Pymes como a grandes empresas. Para reforzar este programa, la compañía ha nombrado una persona responsable del área que controle y organice las certificaciones de las empresas.

Para obtener más información sobre los productos de la compañía, es posible consultar la dirección de Internet: http//www.attachmate.com/europe/spain.

Breves

Las empresas informáticas aumentan sus beneficios. Intela alcanza un alza del 29% para el segundo semestre de 1997

Los nuevos procesadores de Intel, el Pentium con tecnología MMX y el Pentium II, han recibido una acogida rápida en el segundo trimestre, según Intel Corporation.

El Pentium II ofrece sobre la arquitectura Intel los mejores resultados para el cálculo de enteros, cálculo en coma flotante y multimedia. La tecnología MMX completa las necesidades en proyectos multimedia y parece que su combinación ha resultado muy útil para la empresa, a juzgar por los resultados económicos.

BayNetworks ofrece un conmutador de encaminamiento, para el tráfico Intranet/Internet

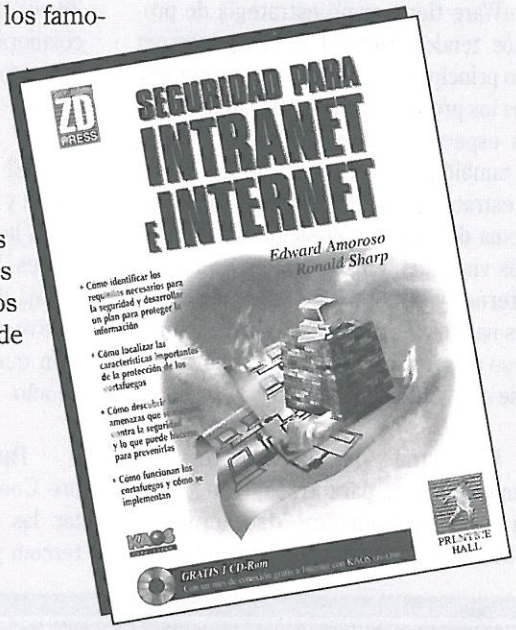
Bay Networks ha presentado su conmutador de encaminamiento Switch Node, que se basa en la arquitectura de conmutación distribuida de Bay Networks y permite envíos IP/IPX a velocidad de cable a latencias de conmutador de nivel 2. Como resultado de acelerar el tráfico IP, el conmutador proporciona una mejor relación precio/prestaciones.

Según la compañía, el conmutador de encaminamiento constituye una solución de centro de red para las redes de tamaño medio que están dedicadas a entornos de campus con mucho tráfico de subred a subred. Asimismo garantiza la interoperatibilidad con cualquier producto de otros fabricantes y puede instalarse en una red existente para proporcionar un incremento inmediato de las prestaciones obtenidas.

LIBROS

SEGURIDAD PARA INTRANET E INTERNET

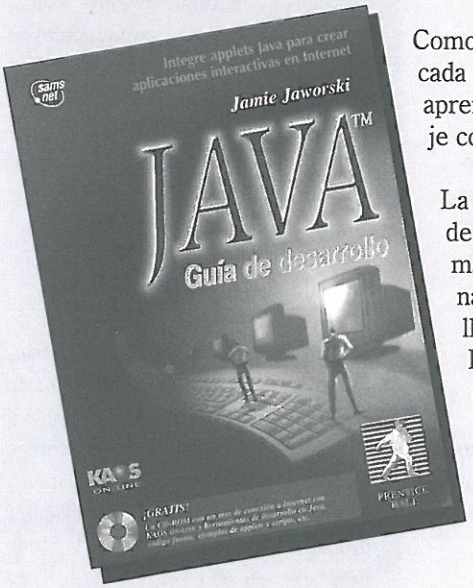
Entendido como una guía práctica para la selección y la utilización de los famosos cortafuegos, Seguridad para Intranet e Internet es una obra que de forma concisa y muy directa invita a conocer en profundidad los sistemas de cortafuegos para intranets y para internet.



El libro ha sido escrito por profesionales desarrolladores de productos para cortafuegos que también han tenido una amplia experiencia como consultores de seguridad en AT&T Bell Laboratories. Sus autores pretenden que se utilice como punto de partida para usuarios que se inician en el conocimiento del diseño y administración de los cortafuegos y no tiene ningún conocimiento sobre ellos. Como punto de partida es una obra muy recomendable.

Editorial: Prentice Hall
Autores: Edward Amoroso y Ronald Sharp
Idioma: Castellano
Número de páginas: 189
Incluye CD-ROM

JAVA GUÍA DE DESARROLLO



Como obra que abarca los principales temas de interés del desarrollo con este cada vez más utilizado lenguaje, Java Guía de desarrollo se puede utilizar para aprender a programar en Java ya que se explican todos los detalles del lenguaje con muchos ejemplos de programación.

La programación de los controles GUI de ventana, la utilización de los sockets de red y de la entrada y salida de datos de forma fluida son algunos de los temas más interesantes que son tratados con fidelidad y claridad. El objetivo final de esta excelente obra que aparece ahora en el mercado, es que el lector llegue a ser un buen programador de Java y, avalada por la experiencia que Prentice Hall tiene en la edición de libros de estas caracterÌsticas, muy bien puede conseguirlo.

Editorial: Prentice Hall
Autor: Jamie Jaworski
Idioma: Castellano
Número de páginas: 687
Incluye CD-ROM

HTML Dinámico con Netscape Communicator y JavaScript (II)

WWW

Alejandro M. Reyero Abad
axl@las.es

El mes pasado aprendimos a utilizar las Style Sheets (Hojas de estilo), una de las novedades que incorpora Netscape Navigator 4.0 (Communicator), más conocidas como HTML Dinámico. Siguiendo con la serie sobre HTML Dinámico, este mes veremos qué son y cómo se utilizan las Layers o capas.

La nueva versión de Netscape Navigator (la 4.0, incluida dentro de la suite Communicator) permite definir capas (*layers*) de contenidos transparentes o translúcidos, teniendo la posibilidad, a su vez de posicionarlos con precisión en nuestras páginas WEB.

Con JavaScript tenemos disponibles muchas operaciones que realizar con las capas: podremos mover *layers*, ocultarlas, cambiar el orden con que se superponen y modificar otros muchos aspectos de la capa. Básicamente, podremos utilizar HTML y JavaScript para crear animaciones dinámicas.

Para determinar cuál es el orden en el que las capas se superpondrán, cada *layer* tendrá asociado un "orden de pila", también conocido como orden Z. El orden Z de una serie de *layers* puede ser numérico, siendo el orden Z del *layer* A igual a 1, el del *layer* B igual a 2, y así sucesivamente.

Las capas pueden ser ncipio de la págopacas. Si la capa es transparente, el contenido de las que están debajo se verá a través de ella (como se ve un fondo a través de un GIF transparente). Se pueden también especificar colores e imágenes de fondo para cada *layer*, de la misma manera que se especifican para el cuerpo de un documento HTML.

En el caso de que la capa en cuestión tenga asociado un color o imagen de fondo, la capa dejará de ser transparente y el contenido de las que estén debajo de ella ya no será visualizado.

Netscape Communicator permite que las layers HTML sean dinámicamente posicionadas de dos maneras diferentes

Indicar también, como final de esta pequeña introducción, que las capas pueden estar "anidadas" dentro de otras, esto es, podremos crear una capa que contendrá otra que a su vez contiene otra más.

Netscape Communicator permite que las capas en HTML sean dinámicamente posicionadas de dos maneras diferentes, que pasaremos a detallar a continuación:

- Utilizando hojas de estilo en cascada (véase número anterior de la revista Solo Programadores).
- Utilizando el TAG <LAYER>.

Utilizando Hojas de Estilo en Cascada

Al igual que para las hojas de estilo en cascada, el W3C ha especificado la utilización de hojas de estilo para posicionar *layers* en el siguiente documento HTML: http://www.w3.org/pub/WWW/TR/WD-positioning. Utilizando la sintaxis de las hojas de estilo en cascada, podemos definir la posición de una layer y darle un nombre dentro del TAG <STYLE>. Por ejemplo, el siguiente TAG <STYLE> define tres *layers*. La primera, llamada *layer1*, estará colocada a cien píxeles de la parte superior de la página y a 150 de la parte izquierda. La segunda *layer* (*layer2*) estará colocado a 150 píxeles de la parte superior y a 250 de la izquierda. Finalmente, la tercera (*layer3*) irá a 250 de la parte superior y a 350 de la izquierda:

```
<HEAD>
<STYLE TYPE="text/css">
<!—
#layer1 {position:absolute;
        top: 100px;
        left:150px;}
#layer2 {position:absolute;
        top: 150px;
        left:250px;}
#layer3 {position:absolute;
        top: 250px;
        left:350px;}
—>
</STYLE>
</HEAD>
```

Aunque las posiciones y los nombres de las *layers* los definimos en el TAG <STYLE>, la especificación de sus contenidos la haremos en el cuerpo de la página. Por ejemplo, en el TAG <SPAN>, podremos especificar el id (identificador) que corresponde a la *layer* deseada:

```
<BODY>
<P>Esto es la página principal.</P>
<SPAN id=layer1>
<H1>Cabecera de Layer 1</H1>
<P>Contenidos de Layer 1</P>
<IMG SRC=images/logo1.gif>
</SPAN>
<SPAN id=layer2>
<P>Aquí va el layer 2</P>
</SPAN>
<H1 id=layer3>
Esto es todo el layer 3.</H1>
</BODY>
```

Utilizando el TAG <LAYER>

El TAG <LAYER> es una manera alternativa para definir *layers*. Este TAG se utiliza en el cuerpo de la página.

Podremos especificar el contenido y la posición de una *layer* dentro del TAG <LAYER>, sin tener que predefinirla con antelación. Hay que hacer notar que, al igual que las hojas de estilo en JavaScript, el TAG <LAYER> es una etiqueta específica del Netscape Communicator.

Tres.htm

```
<HTML>
<HEAD>
<META NAME="GENERATOR" CONTENT="Mozilla/4.0b4 [en] (Win95; I) [Netscape]">
</HEAD>
<BODY>
Esto est&aacute; fuera de los layers
<H2>
Cabecera del Layer 2</H2>
Texto variado incluido en el Layer 2
<BR>
<IMG SRC="layer2.gif" ALT="GIF incluido en el Layer" >
<H1>
Cabecera del Layer 3</H1>
Texto variado incluido en el Layer 3
<H1>
Esta cabecera es todo lo que el Layer 1 contiene</H1>
</BODY>
</HTML>
```

El TAG <LAYER> es una manera alternativa a la de las hojas de estilo en cascada para definir layers

El código HTML de tres.htm es un buen ejemplo de uso del TAG <LAYER>.

Una vez vistas las dos maneras de definir layers, conviene indicar que, independientemente de la que utilicemos, todas las layers tienen reflejo en JavaScript. Podremos escribir scripts JavaScript que modifiquen la capa de diferentes maneras, cambiando la posición, el contenido, la visibilidad, el color, el tamaño. También, mediante JavaScript, podremos realizar animaciones dinámicas directamente en HTML, sin necesidad de utilizar applets Java, ya que las capas se pueden mover, expandir y contraer.

Layers Fijos y Layers de Flujo

Podemos definir dos tipos de *layers*, las que se quedan fijas en una posición en la página y las que siguen el flujo de la página. A las primeros se les conoce como layers fijas, mientras que las segundas son conocidas como layers de flujo.

Las capas fijas necesitan que, como hemos visto, declaremos la posición exacta que ocupan respecto al principio de la página, así como la distancia que hay desde la capa al margen izquierdo, utilizando los atributos *TOP* y *LEFT* respectivamente. Si no definimos estos atributos, la capa tomará una posición por defecto, que será la que determine los valores de *TOP* y *LEFT* de la *layer* en el que se encuentra contenido. Si la *layer* es la "principal", esto es, no

se encuentra dentro de otra, sino que está contenida directamente en la página, se situará en la esquina superior izquierda de la misma.

Las *layers* de flujo aparecen allá donde irían normalmente en el documento HTML, no "sobreescriben" otros elementos HTML. Para crear layers fijas, utilizaremos el TAG <LAYER>, con su correspondiente </LAYER> para cerrarlas. Para crear *layers* de flujo utilizaremos los TAGs <ILAYER> e </ILAYER>. Vemos a continuación un ejemplo de cada tipo de *layer*:

```
<LAYER ID=layer4 TOP=400px LEFT=480px>
<P>Este es el Layer 4</P>
</LAYER>
<ILAYER ID=layer5>
<P> Este es el Layer 5</P>
</LAYER>
```

Si utilizamos hojas de estilo en cascada para definir nuestras capas, necesitaremos definir el tipo de *layer* asignándole un valor a la variable *position*, que será *absolute* si nuestro *layer* es fijo, o *relative* si es de flujo. Por ejemplo:

```
#layer4 {position:absolute;
         top: 400px;
         left: 480px;}
#layer5 {position:relative; }
```

Atributos de las capas

A continuación veremos todos los atributos que podemos definir para los TAGs <LAYER> e <ILAYER>. Utilizaremos la sintaxis <LAYER>, a no ser que se indique lo contrario, el nombre de la propiedad correspondiente es idéntico en la sintaxis de hojas de estilo en cascada.

ID: El parámetro ID se utiliza para especificar el nombre de la capa, para que otras *layers* y JavaScript se puedan referir a ella.

Anteriormente, este atributo se llamaba *NAME*. El atributo *NAME* aún puede ser utilizado, pero se recomienda utilizar *ID* ya que es el mismo atributo que utilizan las hojas de estilo en cascada.
La sintaxis es muy simple:

```
ID=nombre_del_layer
```

LEFT y TOP: Como ya hemos visto, *LEFT y TOP* especifican la distancia que habrá desde la parte superior y el margen izquierdo, respectivamente, hasta la capa. Para las capas fijas, el origen es la esquina superior izquierda del documento o de la capa conteniendo a la capa, incrementándose las coordenadas hacia abajo y hacia la derecha.

Por ejemplo:

```
<LAYER LEFT=100 TOP=200>
<P>Este Layer aparece a 100 pixeles a la derecha
    del margen izquierdo y a 200 pixeles
    por debajo de la parte superior del
    documento</P>
</LAYER>
```

En las *layers* de flujo, el origen es la posición "natural" de la *layer* en el documento, esto es, se comporta como otro elemento HTML cualquiera, sin sobre escribir a otros. Por ejemplo:

```
<P> Este párrafo aparece por encima del layer</P>
<ILAYER LEFT=20 TOP=50>

<P>Este párrafo aparece 20 pixeles a la derecha de
su posición natural, y 50 pixeles de mas, por debajo
del párrafo anterior, de lo que seria normal.</P>

</ILAYER>
<P> Este párrafo aparece por debajo del layer.
```

PAGEX y PAGEY: Estos atributos son exclusivos de la sintaxis <LAYER>, no tienen equivalente en la sintaxis de las hojas de estilo en cascada. *PAGEX y PAGEY* especifican la posición vertical y horizontal de la esquina superior izquierda de la *layer* con respecto al documento en el que se encuentran (en vez de con respecto a la *layer* en el que está contenido nuestro layer). La sintaxis es similar a la de *LEFT y TOP*.

SRC: Indica un fichero externo que contiene código fuente HTML y será visualizado en la *layer*. Este fichero externo podrá contener código JavaScript.

La sintaxis es muy sencilla:

```
SRC="codigo/layers/externo.htm"
```

La utilidad de la propiedad *SRC* es más que evidente, podríamos definir una *layer* dentro de la página WEB de una agencia de viajes con la oferta de la semana. Al finalizar la semana y cambiar la oferta, no tendríamos que modificar todo el código HTML de la página principal, sino únicamente la del fichero que contiene el código de la capa.

SRC indica un fichero externo que contiene código fuente HTML (o JavaScript) y será visualizado en la layer

WIDTH: Determina el tamaño, a lo ancho, de la capa, pudiendo ser expresado en píxeles o también en porcentaje con respecto al tamaño de la página o *layer* que lo contiene (como hacemos en <FRAMESET>).

Hay que hacer notar que los elementos que no puedan ser cortados y que tengan un tamaño más grande que la *layer* en cuestión (como podría ser una foto, por ejemplo), harán que nuestra capa "crezca" hasta adquirir el tamaño necesario.

HEIGHT: Es exactamente igual que el anterior, solo que se utiliza para expresar el tamaño vertical. La principal utilidad de estos dos atributos es utilizarlos como referencia para definir el tamaño de otras *layers* que pudieran estar contenidas en su interior.

CLIP: Este atributo es exclusivo de la sintaxis <LAYER>. Los parámetros de CLIP determinan el rectángulo que contendrá a la *layer*.

La sintaxis es la siguiente:

CLIP=100,60,340,560

Los valores se corresponden con la parte superior, la parte izquierda, la parte inferior y la parte derecha del rectángulo respectivamente. También podríamos indicar únicamente dos valores, asignándose automáticamente el valor cero a la parte superior y a la parte derecha. En el caso de que el atributo *CLIP* fuera omitido, los valores del rectángulo quedarían definidos de forma inmediata por *WIDTH y HEIGHT*.

Z-INDEX, ABOVE y BELOW: *ABOVE y BELOW* no están disponibles en la sintaxis correspondiente a las hojas de estilo en cascada.

El atributo ABOVE especifica la layer que estará justo por encima de la que acabamos de crear

Estos tres valores especifican el orden Z de las capas. Una vez asignado este valor, el orden Z cambia, moviéndose las *layers* acorde con lo establecido. El atributo *Z-INDEX* nos permite definir el orden Z mediante un número entero. Las *layers* con órdenes Z más altos son apilados por encima de los demás. Solamente podremos especificar órdenes Z positivos.

El atributo *ABOVE* especifica la *layer* que estará justo por encima de la que acabamos de crear, esto es, la nueva *layer* es creada justo debajo de la que hemos especificado en *ABOVE*. *BELOW* es exactamente igual que *ABOVE*, solo que especifica la *layer* que irá debajo del que acabamos de crear.

Los valores de *Z-INDEX, ABOVE y BELOW* son relativos a las *layers* que tienen la misma capa padre.

VISIVILITY: Mediante el parámetro *VISIVILITY* se determinará cuándo una *layer* será visible.

Puede tener tres valores:

- *SHOW*, que muestra la capa.
- *HIDDEN*, que la esconde.
- *INHERIT*, que hace que la capa tenga la misma visibilidad que su capa padre.

BGCOLOR: Este parámetro determina el color del fondo de la *layer* (por defecto es transparente). Se especifica igual que el parámetro *BGCOLOR* en el TAG <BODY>, esto es, admite nombre de colores o pares de RGB (#0099FF por ejemplo).

BACKGROUND: Igual que el anterior, sólo que especificamos un URL en el que encontrará la imagen que utilizará de fondo de la *layer*.

OnMouseOver y OnMouseOut: Únicamente se pueden utilizar con la sintaxis <LAYER>.

Son equivalentes a los *handlers* del mismo nombre en JavaScript, el primero es invocado cuando el puntero del ratón entra en la *layer*, y el segundo en el momento en que sale.

OnFocus y OnBlur: Iguales que los anteriores. El primero es invocado cuando la *layer* tiene el foco del teclado, y el segundo cuando lo pierde.

OnLoad: Solamente se pueden utilizar con la sintaxis <LAYER>. Es otro *handler* que se invoca cuando la *layer* es cargada, independientemente de si tiene el primer plano o no. Su sintaxis es:
OnLoad="haz_algo; return false;"

<NOLAYER>: Es un TAG HTML. Si el browser que el cliente utiliza para ver nuestra página no soporta *layers*, presentará el contenido incluido entre <NOLAYER> y </NOLAYER>. Es similar a <NOFRAME>.

Como hemos visto anteriormente, podemos utilizar JavaScript para crear dinámicamente y modificar *layers*.

El objeto Layer

Para cada *layer* en el documento, ya sea creada con el TAG <LAYER> o mediante hojas de estilo en cascada, existirá un objeto *layer* correspondiente en JavaScript. Las propiedades del objeto *layer* son un reflejo de las propiedades de la capa en nuestra página. Algunas de estas propiedades podrán ser modificadas por el usuario mediante JavaScript, otras no.

JavaScript puede ser utilizado para crear y modificar layers dinámicamente

Ya que las *layers* son objetos JavaScript, podremos acceder a sus métodos y propiedades de la siguiente manera:

```
nombreLayer.nombreMetodo(parametros)
nombreLayer.nombrePropiedad
```

La Propiedad Document del objeto layer

Cada documento tiene una propiedad *layers* que contiene un array con todas las layers principales de ese documento

(esto es, las *layers* padre). A su vez, cada layer padre tiene una propiedad *document*, que a su vez tiene una propiedad *layers* que contiene todas las *layers* principales incluidos dentro de la *layer* padre.

La propiedad *document* de cada *layer* contiene a su vez todas las propiedades comunes de un documento, como pueden ser las propiedades *images*, que contiene todas las imágenes del layer, *applets* que contiene un array con todos los applets de un layer, etc.

Esto que, sobre el papel, puede que no quede lo suficientemente claro, lo vemos con un ejemplo:

```
<HTML>
<BODY>

<LAYER ID = "LAYER1">
    <P>Esto es el Layer 1</P>
</LAYER>

<LAYER ID = "LAYER2">
    <P>Esto es el Layer 2</P>
    <LAYER ID = "LAYER21">
        <P>Esto es el Layer 2.1</P>
    </LAYER>
</LAYER>

</BODY>
</HTML>
```

Las siguientes líneas de código JavaScript ilustran el uso de las propiedades *layer* y *document*:

```
document.layers //su valor es un array de 2 layers
document.layers["layer1"] //su valor es layer1
document.layers["layer2"] //su valor es layer2
document.layers["layer21"] //Produce un error, ya
    //que layer21 no es un layer //
principal.document.layers["layer2"].document.layers
                                        ["layer21"]
    //su valor es layer21
```

Mediante la utilización de JavaScript, podremos acceder a objetos en el array *layers* a través de su número índice (que tendrá que ser un entero no-negativo), o también, utilizando el nombre que le asignamos con el atributo ID en el TAG <LAYER>.

Cuando accedemos mediante su número índice, los elementos del array aparecen ordenados por su orden Z, empezando desde atrás a adelante, donde la cero es la *layer* que está mas abajo en la pila de *layers*, incrementándose el número a medida que vamos subiendo en esa pila.

El índice de una *layer* no es igual a su número de orden Z, ya que en este último no es necesario que los números sean consecutivos.

Los siguientes fragmentos de código son ejemplos de accesos validos a objetos *layer*:

```
document.layers[2]
document.layers["layer3"]
document.layers["layer3"].document.layers
                                        ["layer34"]
```

Escribe.htm

```
<HTML>
<BODY>
<ILAYER                    ID="layer1"
BGCOLOR="green">
 <HR>
 <P>Este es el contenido original.<P>
 <HR>
</ILAYER>
</BODY>
</HTML>
<SCRIPT>
 function modificaContenido() {

document.layers["layer1"].document.wri-
te("<HR>
<P>Nuevo Contenido.</P><HR>");

document.layers["layer1"].document.clo-
se();
}
</SCRIPT>

<FORM NAME="formLayer">
 <INPUT TYPE=button
VALUE="Modifica Contenido"
OnClick='modificaContenido();return
false;'>
</FORM>
```

Los elementos de un array *layers* no pueden ser asignados "a mano". Esto sigignifica que por ejemplo la siguiente asignación dentro de código JavaScript es errónea y será ignorada:

```
document.layers[1]="layer5"
```

Aún así, las propiedades de los objetos en el array pueden ser leídas, y, algunas, modificadas. Por ejemplo:

```
document.layers[2].top=200
```

Creando Layers Dinámicamente

Se pueden crear nuevos objetos *Layer* utilizando el operador *new* sobre un objeto *layer*.

Por ejemplo:

```
layer2=document.layers["layer2"]
layerNuevo=new Layer(500, layer2)
```

El primer argumento es la anchura de la capa que acabamos de crear, y el segundo es su padre, si no se lo indicamos, su padre será el documento principal. Después de crear la *layer*, podremos indicarle su código fuente dándole un valor a su propiedad SRC, o con el método *load*.

Escribiendo en Layers

Una vez que la *layer* ha sido creada y la página se ha cargado completamente, tendremos la posibilidad de modificar su contenido utilizando el método *write()*. Si utilizamos el método *write* para escribir en el documento, tendremos que ser concientes de que el contenido original del layer desaparecerá, siendo entonces sustituido por el nuevo.

El método write sirve para modificar el contenido de una página ya creada y cargada

También conviene recordar que una vez escrito en una *layer* tendremos que cerrar el documento. Como ejemplo podemos observar el código fuente de escribe.htm

En el interior de una *layer* podremos utilizar los TAGs <SCRIPT> y </SCRIPT>.

Las funciones JavaScript que creemos quedarán reducidas únicamente al ámbito de la *layer* en la que éstas se incluyen, no pudiendo ser utilizadas fuera de ella.

Animando Layers

Mediante la función JavaScript *setTimeout* es posible animar *layers*. Esta función llama a otra función una vez que ha transcurrido un determinado tiempo que se indica como parámetro.

La función *setTimeout* puede ser invocada de dos maneras diferentes:

- setTimeout("Código a ser ejecutado", tiempo)
- setTimeout(función, tiempo, argumentos...)

Por ejemplo, para llamar a la función *estamosProbando*, pasándole dos argumentos y haciendo que espere 10 milisegundos:

```
setTimeout("estamosProbando('arg1','arg2')",10)
setTimeout("estamosProbando", 10, "arg1",
                              "arg2")
```

El siguiente ejemplo hace que una *layer* crezca de 5 en 5 hasta llegar a los 450 píxeles:

```
function expand(layer)
{
    if (layer.clip.right < 450)
    {
        layer.resizeBy(5, 5);
        setTimeout("expand", 5, layer);
    }
    return false;
}
```

También podremos animar imágenes mediante layers, cambiando la fuente de una imagen en conjunción con el movimiento de la imagen.

Para mover una imagen, tendremos que cambiar la posición de la layer de la imagen

Para mover la imagen, únicamente tendremos que cambiar la posición de la *layer* de la imagen. Para cambiar la imagen, podremos asignar un nuevo valor a la propiedad SRC de la imagen. Si el SRC de la imagen es cambiado demasiado deprisa, la imagen actual puede que no llegue de la red a la velocidad deseada para hacer que la animación sea fluida, en este caso lo mejor es no comenzar la animación hasta que la imagen no haya sido completamente cargada.

Conclusiones

Una vez vistas las *layers*, y en el número anterior las hojas de estilo, estamos preparados para hacer páginas totalmente compatibles con las nuevas opciones de Netscape Communicator, que confieren una gran potencia a nuestras creaciones.

En el momento en el que leáis esto, Microsoft debería haber sacado la versión 4.0 de su Internet Explorer (o por lo menos una Beta casi definitiva). Esperemos que dejen su orgullo a un lado e incluyan las nuevas extensiones al lenguaje HTML de una manera 100% compatible con Netscape.

Bibliografía

- "JavaScript: Una chispa de vida WEB", por Fernando J. Echevarrieta. Sólo Programadores número 30.
- "El Modelo de Objetos de JavaScript", por Alejandro M. Reyero. Sólo Programadores número 32.
- "JavaScript: Frames y Windows", por Alejandro M. Reyero. Sólo Programadores número 33.
- "LiveConnect, El Puente Entre Java y JavaScript", por Alejandro M. Reyero. Sólo Programadores número 34.
- "HTML Dinámico con Netscape Communicator y JavaScript", por Alejandro M. Reyero. Sólo Programadores número 35.

Contactar con el autor

Si el lector quiere hacer llegar sus opiniones, preguntas sobre los art;iculos que componen la serie, ejemplos, o, simplemente, decir "hola" al autor puede hacerlo a través de las siguiente dirección de correo electrónico:

E-Mail: axl@las.es
o bien consultando la dirección WEB:

http://www1.las.es/~axl.

NÚMERO EXTRA

SÓLO PROGRAMADORES

LINUX

Debian 1.3.1

VERSIÓN OFICIAL en 2 CD-ROM

2 CD-ROM + MANUAL DE INSTALACIÓN POR SÓLO 2.495 Ptas.

MUY PRONTO EN TU QUIOSCO

INCLUYE:

- 974 paquetes de Software.
- Miles de páginas de texto en formato HTML.

CARACTERÍSTICAS PRINCIPALES:

- Han intervenido en esta distribución más de 200. desarrolladores (el mayor grupo Linux).
- Es compatible Slackware y RPM (Red Hat).
- Todo el software incluido ha sido testeado por un excepcional pre-release testing group.

CON LA GARANTÍA DE UNA DISTRIBUCIÓN OFICIAL

TOWER

c/ Aragoneses, 7 - 28108 Pol. Ind. Alcobendas (Madrid) - Tel.: (91) 661 42 11* - Fax: (91) 661 43 86
e-mail: solop@towercom.es http://www.towercom.es

Facturación en Novell Netware

Enrique Díaz Trobo

El sistema de facturación de Novell Netware no sólo es útil para cobrar por servicios que ofrecemos. También puede ser muy útil para ver el uso que se hace de la red y para repartir costes de producción entre varios departamentos en una empresa. En este artículo veremos sus aplicaciones y la forma de instalarlo paso a paso.

Aunque la facturación se asocia al cobro a los usuarios por el tiempo y los recursos utilizados en la red, el sistema de facturación de Netware también ofrece información de utilidad sobre las conexiones y desconexiones de los usuarios. Los administradores pueden realizar un seguimiento de los registros de utilización, de forma que es posible determinar con exactitud los recursos que están siendo usados y por qué usuarios. Esta información es de utilidad cuando se necesita determinar si son necesarios equipos y sistemas de almacenamiento adicionales, y para justificar su compra ante la empresa.

El sistema de facturación de Netware se instala en el nivel de servidor, de forma que se puede evaluar el uso que se hace de un servidor en particular y facturarlo si es necesario. Así, por ejemplo, se podría saber en qué medida ha contribuido un departamento en la realización de un proyecto viendo el uso que ha hecho de recursos, como es el almacenamiento en disco, a efectos de presupuesto. También puede que un departamento necesite hacer un seguimiento de la distribución de gastos en un contrato. Las instituciones educativas y los servicios a tiempo compartido necesitan hacer a menudo un seguimiento del uso de los recursos del sistema y facturarlo según el tiempo a los usuarios.

La utilidad de facturación usa un sistema por puntos para contar el tiempo de trabajo o el uso de los recursos por parte de los usuarios. A estos puntos se les puede asignar un valor monetario, o también tratarlos como vales o crédito que los usuarios pueden gastar. Un administrador puede asignar a un usuario temporal un cierto número de puntos. Cuando gasta sus puntos, el usuario puede pedir más y de esta forma el supervisor conoce el tiempo y los recursos que utiliza este usuario temporal. Estos puntos pueden estar limitados o no, según las necesidades de gestión del sistema.

Un ejemplo práctico

En un entorno educativo, a los estudiantes se les puede asignar una cierta cantidad de recursos y facturarles por los recursos utilizados. Periódicamente podemos generar un informe que sería la base de tarifas, según el uso. A la inversa, los estudiantes, al principio del periodo escolar podrían comprar un bono por un tiempo de uso y espacio en disco. Al gastar el bono, deberán comprar más. Así, los estudiantes que fueran conscientes de sus límites serían más productivos economizando en el uso de la red.

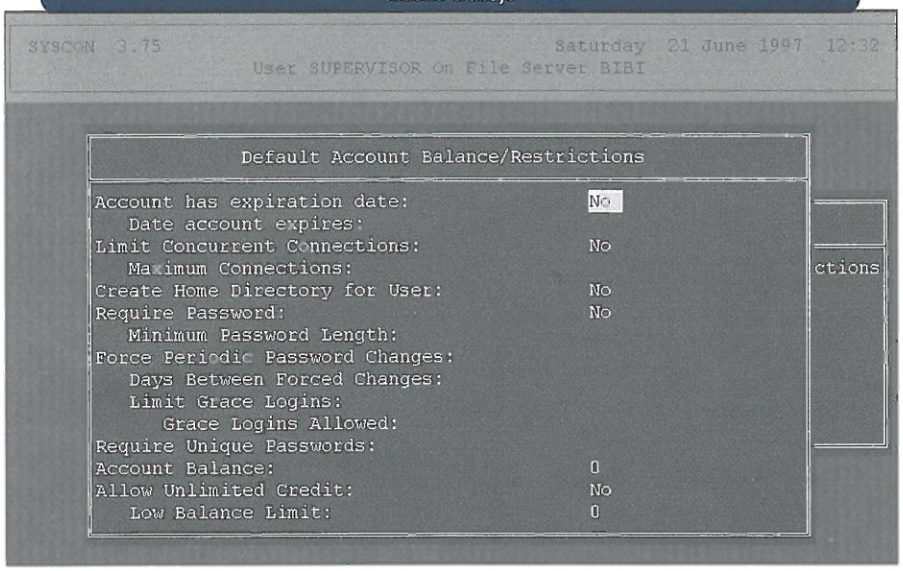
El primer paso para hacer efectivo el sistema de facturación es fijar las tarifas por el uso de las cuentas; la mejor estrategia es establecer un periodo de prueba,

instalando la utilidad de facturación y haciendo un seguimiento del uso del sistema, dándole a los usuarios un crédito ilimitado. Al final del periodo de prueba, se dispondrá de la suficiente información para fijar las tarifas.

El sistema de facturación se instala por separado en cada servidor de la red. Si no se necesita en un servidor no se debe instalar, ya que ocupa espacio en disco y gasta tiempo de procesamiento.

La utilidad ATOTAL nos ofrece información sobre la facturación, resume la información y genera una lista con los tiempos de conexión, peticiones de servicio, lecturas y escrituras de bloques y requisitos de almacenamiento en disco.

Establecer unos valores por defecto de los datos de facturación nos puede ahorrar mucho trabajo



Tipos de facturación

El seguimiento de las conexiones y desconexiones al sistema de los usuarios se activa automáticamente al instalar la utilidad de facturación. A cada usuario se le asigna una cuenta de cargos, donde se anota el uso efectuado de los recursos. Todo servidor de archivos en el que se definan cargos genera apuntes en las cuentas de los usuarios.

A los usuarios se les puede imputar dos tipos de cargos principales:

- Cargos por el uso del espacio en disco del servidor de archivos.
- Cargos por los trabajos ejecutados por el servidor.

Si se va a facturar el uso de espacio en disco será necesario fijar una tarifa y determinar la frecuencia y los momentos en que el servidor va a evaluar el espacio en disco ocupado por el usuario. Los cargos generados por los trabajos realizados por un servidor se listan a continuación, y se puede aplicar uno o varios de estos cargos:

- La cantidad de recursos consumidos no sólo de los servidores de archivos, sino también en los gateways, servidores de impresión y servidores de base de datos.
- Cantidad de tiempo que el usuario está conectado al servidor.
- Cantidad de información leída del disco del servidor.
- Cantidad de información escrita en el disco del servidor.
- El número de peticiones de operación hechas al servidor.

Instalación del sistema de facturación

Antes de poder hacer un seguimiento de las conexiones y desconexiones de los usuarios, y de que se pueda facturar el uso de los recursos, se ha de instalar la utilidad de facturación. Para instalar el sistema se puede utilizar Netware Administrator o la utilidad NETADMIN o

Figura 1. Instalación de la configuración paso a paso. Al seleccionar la opción Accounting se nos pedirá confirmación para instalar el sistema de facturación.

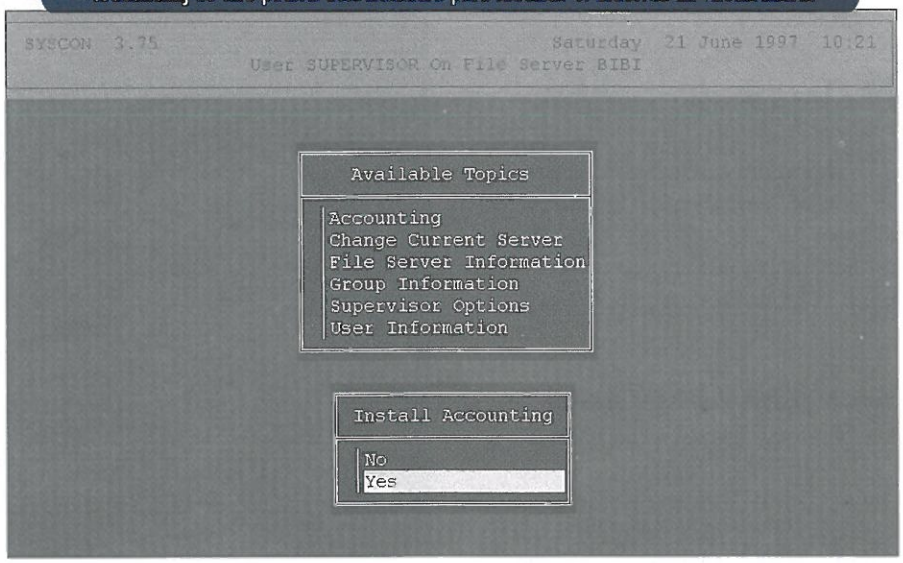
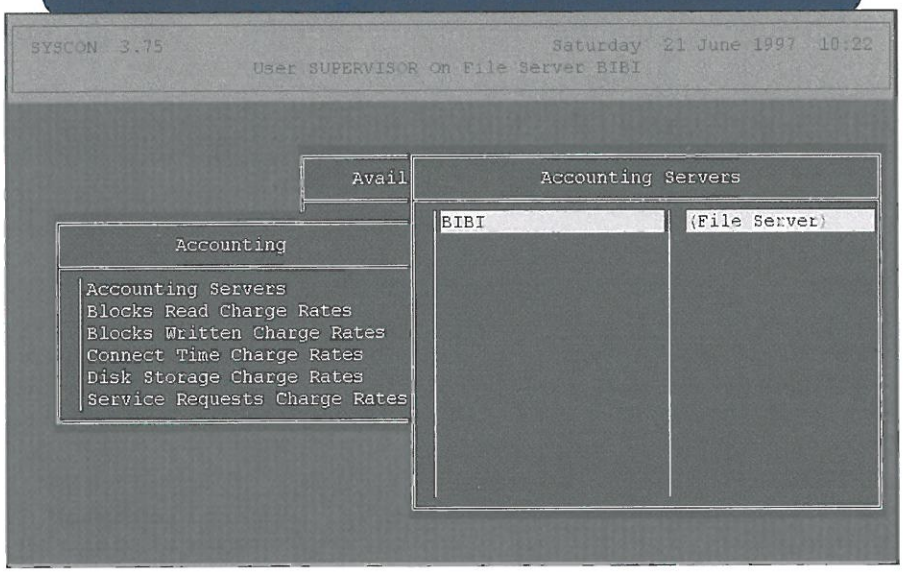


Figura 2. Después deberemos elegir los servidores que tendrán sistema de Facturación.



también SYSCON para las versiones 3 de Novell Netware. Para instalar y utilizar los servicios de facturación con NETADMIN seleccionamos *Manage objects* (gestión de objetos) en el menú principal de NETADMIN. Examinaremos el árbol de directorios y seleccionamos el servidor en el que deseamos instalar el sistema de facturación. Seleccionamos *Yes* (sí) cuando pida confirmación sobre la instalación de las opciones de facturación. Aparecerá un nuevo cuadro del que seleccionaremos la opción *Accounting Servers* (Servidores de facturación), para seleccionar el servidor con el que trabajar.

Tipos de tarifas

Hay varias categorías de tarifas que se pueden utilizar para facturar a los usuarios los servicios ofrecidos por el servidor de la red. Cada opción se describe en las siguientes secciones:

Blocks Read Charge Rates (tarifas por lectura de bloques).

Se utiliza para fijar las tarifas de facturación sobre la cantidad de información leída en el servidor. No es lo mismo que facturar los bloques almacenados en el disco; esto se verá más adelante con la opción *Disk Storage Charge Rates* (tarifas para almacenamiento en el disco). Se asigna una tarifa en incrementos de media hora y por cada bloque leído, siendo el tamaño de bloque de 4.096 *bytes*.

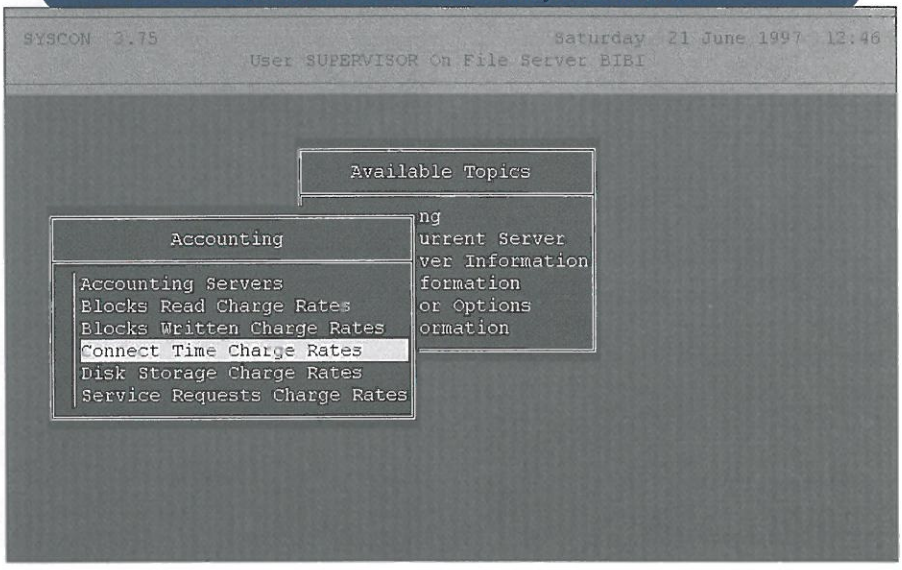
La cantidad de bloques leídos puede llevar a confusión, ya que cada vez que se abre una base de datos y se lee, se añaden los cargos, incluso en el caso de que no se realice ningún trabajo. Cada vez que el usuario lee un bloque se realiza un cargo en su cuenta. La utilización de esta opción es importante en los sistemas con servidores que suministran información a los usuarios, como los sistemas en línea o las bases de datos. Hay que tener en cuenta que esta opción puede hacer que caiga la productividad o la calidad del trabajo, ya que si temen consumir demasiado es posible que no utilicen todas las herramientas que el sistema pueda proporcionarles.

Los cargos por tiempo de conexión no resultan recomendables si hay muchos usuarios noveles

Blocks Written Charge Rates (tarifas por escritura de bloques).

Similar a la anterior, excepto en que realiza el seguimiento de la información

Figura 3. A continuación elegiremos la opción Connect Time Charge Rates para establecer una tarifa sobre el tiempo de conexión.



que los usuarios escriben en el disco, en lugar de la que leen. Las tarifas por escritura de bloques no son las mismas que las provenientes del almacenamiento en disco (*Disk Storage Charge Rates).* Se asigna una tarifa por cada bloque de 4.096 *bytes* escrito, pudiendo darse un valor distinto para cada intervalo de media hora.

Cada vez que se escribe un bloque en el disco, se hace un cargo en la cuenta del usuario. Hay que ser cuidadoso al utilizar esta opción, ya que algunos programas escriben continuamente en el disco, penalizándose en este caso a los usuarios de estos programas. Por ejemplo, cuando se instala Windows en el servidor de archivos, este escribe archivos temporales en el servidor y los actualiza de forma regular. Al igual que en el caso de las tarifas por lectura del disco, puede ocurrir que la productividad de los usuarios disminuya si saben que se les pasan cargos por escribir en el disco.

ConnectTime Charge Rates (tarifas por tiempo de conexión).

Factura el tiempo que los usuarios están conectados al servidor. El tipo de tarifa se puede establecer por periodos de media hora y la unidad de facturación es el minuto. Sin embargo antes de aplicar este método es importante analizar el tipo de usuario y los recursos disponibles en el sistema. Si el uso de la red es grande, puede ser interesante definir tarifas elevadas, favoreciendo que los usuarios sean más eficientes con los trabajos que realizan con el sistema, no acaparándolo mucho. Sin embargo, en algunos sistemas esto no es lo adecuado, especialmente en sistemas recién instalados o si hay muchos usuarios noveles.

Disk Storage Charge Rates (tarifas por almacenamiento en disco).

Realiza un cargo por cada bloque de disco ocupado. Los bloques son de 4.096 *bytes* o 4K. Se establece una tarifa para los incrementos del número de bloques en intervalos de media hora; este incremento se asigna en base a un número de bloques/día, que corresponde al número de bloques almacenados en un día. Si la red posee una capacidad de almacenamiento limitada, los cargos pueden forzar a los usuarios a ser más eficientes en el uso del espacio, y mantener su área de almacenamiento libre de archivos innecesarios.

Service Request Charge Rates (tarifas por utilización del servidor).

Se utiliza para establecer cargos por el uso en general del servidor. Cada vez que un usuario envía una petición al servidor para cualquier operación, se hace un cargo. Las tarifas se pueden definir para cada intervalo de media hora, facturándose al usuario por petición recibida. Se hacen cargos desde el momento de conexión hasta el de desconexión.

Este tipo de facturación incluirá muchos registros en el archivo de facturación, ya que cualquier actividad genera peticiones de servicios. Deberemos estar atentos al tamaño del archivo de facturación si se utiliza esta opción.

Especificación de las tarifas

Es necesario determinar los tipos de cargos y las tarifas con que se desea facturar a los usuarios por los servicios que les ofrece el servidor de red. Así, se debe fijar un balance inicial y ver la información sobre facturación, bien como ayuda en la fijación de las tarifas iniciales, o bien para facturar el uso realizado.

Las tarifas de facturación se establecen fijando un valor numerador y otro denominador. Como numerador se tiene el precio de facturación, normalmente pesetas, y el divisor es el número de unidades que corresponde al precio, normalmente una. Además, podemos tener hasta veinte tarifas, de forma que es posible tener una tarifa diferente en momentos distintos del día. Por ejemplo, se podría establecer una tarifa de conexión superior para periodos de utilización de día que por la noche.

Figura 4. Para definir una tarifa marcaremos con F5 el bloque de tiempo al cual será aplicable la tarifa. Al Pulsar Enter nos pedirá la tarifa aplicable a ese bloque. En este caso elegimos la opción Other Charge Rate para definir la nueva tarifa.

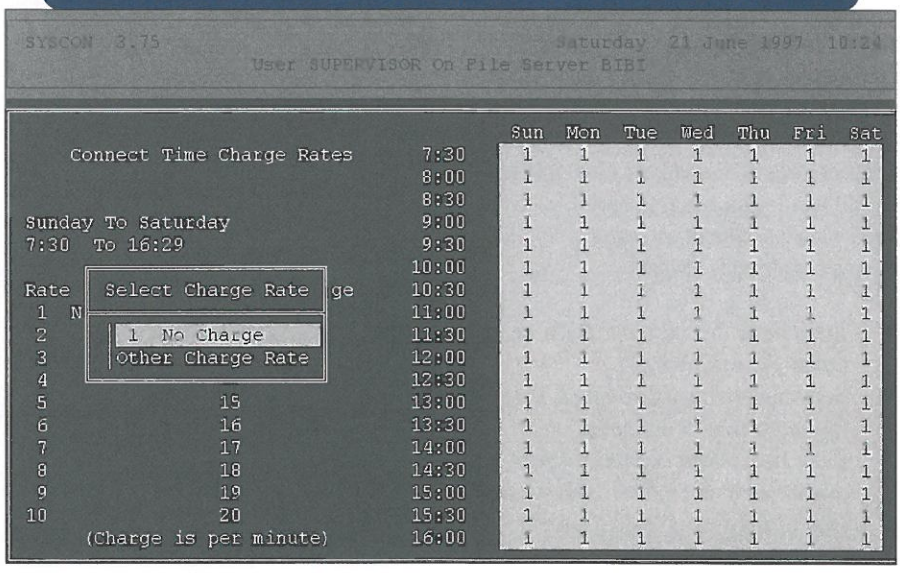
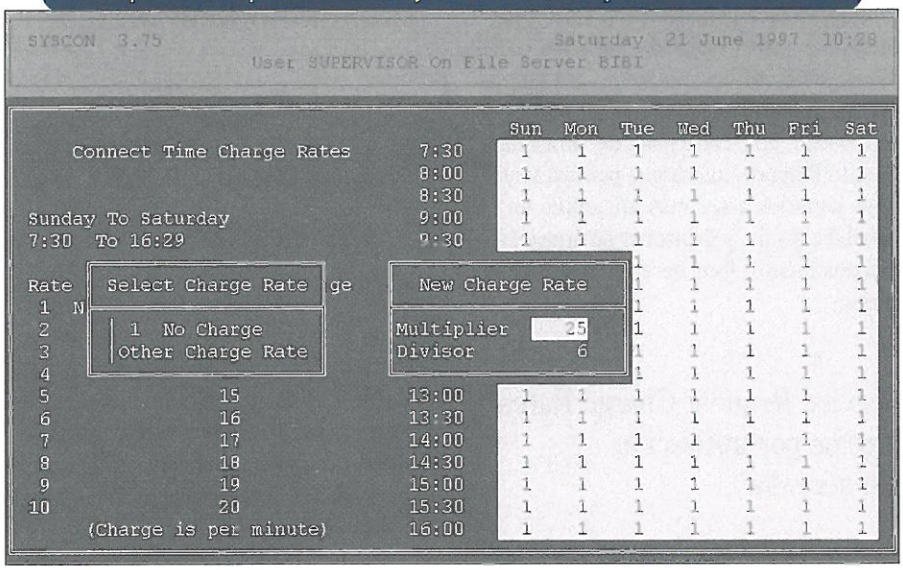


Figura 5. Pretendemos establecer una tarifa de 250 pesetas por hora de conexión. Así pues el multiplicador será 250 y el divisor 60, o lo que es lo mismo, 25/6.


SYSCON 3.75
Saturday 21 June 1997 10:28
User SUPERVISOR On File Server BIBI
Connect Time Charge Rates
Sunday To Saturday
7:30 To 16:29
Sun Mon Tue Wed Thu Fri Sat
7:30 1 1 1 1 1 1 1
8:00 1 1 1 1 1 1 1
8:30 1 1 1 1 1 1 1
9:00 1 1 1 1 1 1 1
9:30 1 1 1 1 1 1 1
Rate
1 N
2
3
4
5 15 13:00 1 1 1 1 1 1 1
6 16 13:30 1 1 1 1 1 1 1
7 17 14:00 1 1 1 1 1 1 1
8 18 14:30 1 1 1 1 1 1 1
9 19 15:00 1 1 1 1 1 1 1
10 20 15:30 1 1 1 1 1 1 1
(Charge is per minute) 16:00 1 1 1 1 1 1 1
Select Charge Rate
1 No Charge
Other Charge Rate
New Charge Rate
Multiplier 25
Divisor 6

Las tarifas se fijan en un proceso con dos pasos:

- Crear una o más tarifas
- Seleccionar un bloque de tiempo y a continuación especificar la tarifa que tendrá efecto en ese periodo.

Las cuotas iniciales de las cuentas de los usuarios se fijan a través de sus cuadros de diálogo *Account Balance*. En Netware Administrator se marca (pinchando dos veces) el objeto usuario, y a continuación se selecciona el botón *Account Balance* (cuota) del cuadro de diálogo.

Con NETADMIN se selecciona el objeto usuario, donde se elige *View or Edit properties of this object* (Ver o editar propiedades del objeto) en el menú visualizado. Seleccionamos *Account restrictions* (restricciones de cuenta) en el menú *View or Edit user* (ver o editar usuario), y a continuación seleccionamos *Account balance*. Después podremos realizar:

- Especificar una cuota limitada en el campo *Account Balance*.
- Seleccionar *Allow Unlimited Credit* (permitir crédito ilimitado)
- Desactivar *Allow Unlimited Credit* y asignar un límite inferior, a partir del cual el usuario no podrá acceder a los recursos de la red.

Para realizar un seguimiento del uso del sistema por parte de cada usuario, asignamos crédito ilimitado. No debemos olvidar que los usuarios serán desconectados de la red cuando alcancen el límite de su cuota, por lo que tendremos que concederles nuevas cuotas, lo cual nos puede suponer mucho trabajo. Asignando crédito ilimitado se generarán apuntes referentes al uso de los recursos de la red por parte del usuario y no tendremos que concederle crédito para ello.

La cuota se especifica en unidades. Un crédito limitado se asigna especificando *No* en el campo *Allow Unlimited Credit* y escribiendo un valor en el campo *Low Balance Limit* (límite inferior del crédito). Si se da un número negativo, los usuarios recibirán servicio hasta el límite de facturación. Si se introduce un valor positivo en el campo, el usuario debe mantener algún valor en el campo. Los usuarios siempre pueden dirigirse al supervisor y pedirle servicios adicionales.

Las cuotas iniciales también se pueden asignar a las cuentas de usuario por omisión en cada contenedor.

Asignar las tarifas

Para establecer una nueva tarifa marcamos sobre el botón *Add Charge Rate* (añadir tarifa). Aparecerá entonces el cuadro de diálogo del mismo nombre que muestra los valores de numerador *(Multiplier)* y denominador *(Divisor)*.

Figura 6. Una vez establecida la tarifa debemos establecer unos créditos para los usuarios. En este caso otorgamos un crédito de 3.000 puntos, no concedemos crédito ilimitado, pero permitimos que sobrepase su crédito en otros 3.000 puntos.

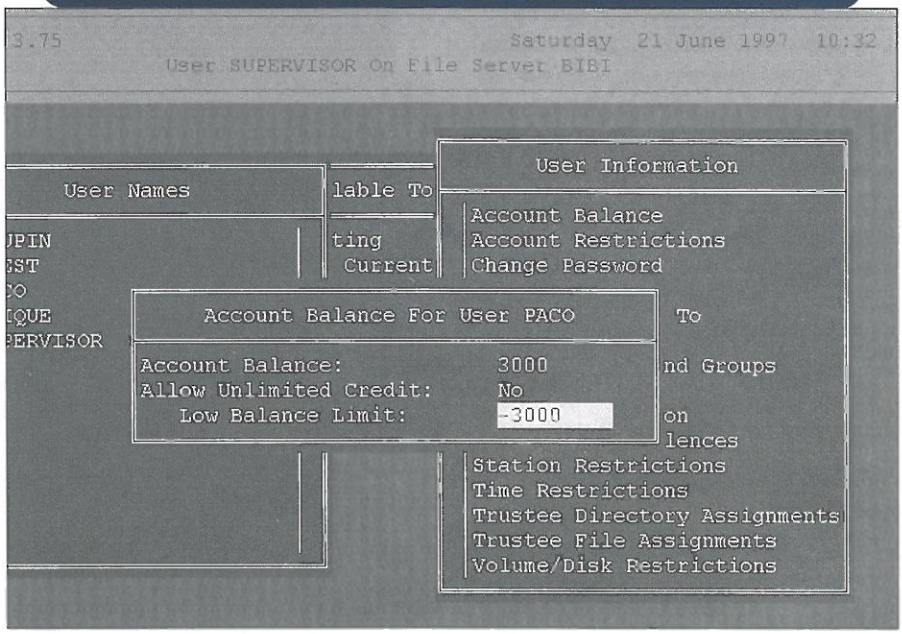


Figura 8. La utilidad ATOTAL nos presentará un informe de los totales facturados en el servidor.

```
F:\SYSTEM>atotal

Accouting Services Total Utility

06/21/1997:
    Connect time:          72     Server requests:      31020
    Blocks read:         3528     Blocks written:        3052
    Blocks/day:

Totals for week:
    Connect time:          72     Server requests:      31020
    Blocks read:         3528     Blocks written:        3052
    Blocks/day:
```

Si deseamos modificar una tarifa debemos borrarla y volver a crearla. Esto supone un cierto problema, ya que no se puede borrar una tarifa si hay un bloque de tiempo que la utiliza. Primero se han de cancelar todos los bloques de tiempo que usen la tarifa, ya continuación se destaca ésta y se pulsa el botón *Delete Charge Rate* (eliminar tarifa). Seguidamente crearemos la nueva tarifa.

Los usuarios serán desconectados cuando alcancen el límite de su cuota y podrán volver a conectarse cuando obtengan nuevo crédito

Seleccionamos una de las tarifas que hayamos creado. A continuación, en la parte superior del cuadro de diálogo, pulsamos y arrastramos el bloque de tiempo al que se desea asignar a la tarifa. A medida que se mueve el ratón se va formando un cuadro sobre el bloque de tiempo. Al liberar el botón del ratón, el bloque de tiempo tomará el color asignado a la tarifa.

Si lo hacemos con NETADMIN, en el lado izquierdo de la pantalla aparecen las veinte tarifas posibles. En el lado derecho se encuentran los bloques de tiempo, que en principio tienen asignada la tarifa uno que ya está establecido por defecto en *No Charge* (sin cargo).

Para asignar una tarifa a una hora específica del día o de la semana, pulsamos F5 (marcar), y con las teclas de cursor destacaremos el área a la que se va a asignar una tarifa específica. Una vez que se ha destacado el área total, pulsaremos INTRO. Aparecerá entonces un cuadro *Select Charge Rate* (Seleccionar tarifa).

Además es posible seleccionar una de las tarifas existentes destacando una de las opciones, o también definir una nueva tarifa seleccionando *Other Charge Rate* (otra tarifa) en la pantalla.

Con las teclas de cursor se destacan tanto el numerador como el denominador, dándose la fracción relativa a un servicio concreto. Esta tarifa se aplica al bloque destacado.

Cómo calcular tarifas

Se pueden establecer tarifas distintas en cada servidor de red, pero el administrador de ésta debe asegurarse de que todos los sistemas utilicen el mismo método para fijarlas. De esta forma se asegura uniformidad y ecuanimidad cuando un departamento le factura a otro el uso de sus recursos.

Para empezar, es una buena idea utilizar el sistema de facturación para hacer un seguimiento del uso del servidor. De esta forma se tendrá una idea de cómo facturar los servicios. Una vez se ha hecho la prueba durante una o dos semanas, se puede utilizar la utilidad ATOTAL para obtener la información sobre la utilización y fijar las tarifas.

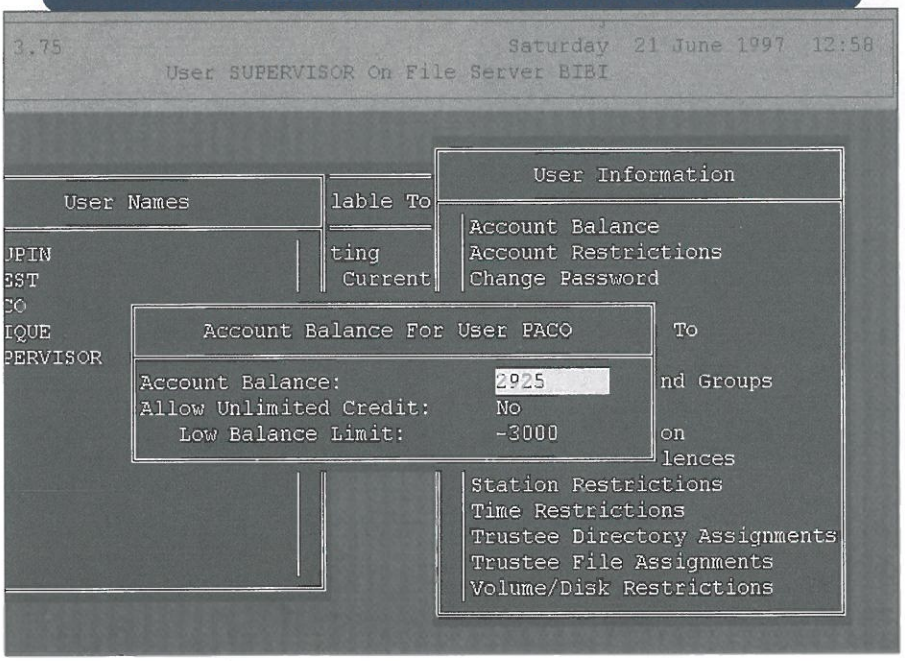
La información del sistema de facturación se almacena en el archivo llamado NET$ACCT.DAT, que puede llegar a ser bastante grande, por lo que es necesario obtener periódicamente una copia impresa de su contenido, o resumirlo con una aplicación de facturación adecuada (existen varias en el mercado) y borrarlo a continuación.

En el momento en que se conecte a la red un nuevo usuario, el sistema de facturación de la misma volverá a crear un nuevo archivo.

Figura 7. Una vez agotado el crédito, el sistema no permitirá la entrada al usuario hasta que le sea concedido nuevo crédito.

```
F:\SYSTEM>login grupin
Enter your password:
BIBI/GRUPIN: You have exceeded your credit limit for this server.
You are attached to server BIBI.


F:\>
```

Figura 9. Podremos consultar el estado del crédito de un usuario seleccionando Account Balance en la información sobre el usuario.



Las tarifas (en forma de fracción) se calculan según esta fórmula:

Numerador de la Tarifa/ Divisor de la Tarifa

El numerador es la cantidad de dinero que desea facturar por un tipo de servicio, como puede ser la lectura de bloques. El denominador es normalmente uno, por lo que indica una unidad de servicio. Esta fracción se asigna seguidamente a períodos específicos del día, pudiendo utilizarse hasta veinte fracciones. De esta forma, por ejemplo, los cargos pueden ser mayores durante el día que durante la noche.

Tarifa por bloques leídos	Cargo por bloque
Tarifa por bloques escritos	Cargo por bloque
Tarifa por tiempo de conexión	Cargo por minuto
Tarifa de almacenamiento en disco	Cargo por bloque-día
Tarifa por petición de servicios	Cargo por petición recibida

Las tarifas se muestran para cada media hora en un período semanal, utilizando los modos de facturación que aparecen listados en la tabla mostrada en la parte superior.

Supongamos que se desea facturar tres pesetas por cada unidad de servicio pedido o utilizado. El numerador es 3 y el denominador es 1, quedando 3/1. Para esta tarifa, se escribiría 3 en el campo *Multiplier* y 1 en *Divisor.*

No es posible facturar precios decimales, pero se puede conseguir expresándolos en formato de quebrado. Supongamos que se desea facturar 1,50 pesetas por unidad de servicio. No podemos poner el decimal, por lo que se multiplica el valor por 100, convirtiéndose en 150. Por ello será necesario multiplicar también el denominador por 100, con lo que queda 100. La tarifa será 150/100, por lo que se tendrá que escribir 150 en el campo *Multiplier* y 100 en el campo *Divisor.*

Otra forma de decidir la tarifa de los servicios es definir la cantidad de dinero que deseamos recaudar por un servicio. Supongamos que en el período inicial de prueba hemos visto que un departamento usualmente emplea unos 300.000 bloques para los que se estima un coste 100.000 pts. que queremos aplicar a efectos presupuestarios. De esta manera el numerador sería 100.000 y los 300.000 bloques de espacio en disco se convierten en el valor de la unidad, y por tanto en el divisor, así: 100.000/300.000 = 1/3.

Para iniciar un seguimiento del uso de los recursos, estableceremos una tarifa 1/1 y eliminaremos las restricciones de cuota

Si únicamente deseamos hacer un seguimiento, entonces fijaremos una tarifa para cada categoría sobre la que se desea información, por ejemplo 1/1, y nos aseguraremos de que no hay restricciones de cuota.

Para calcular los totales de facturación emplearemos la utilidad ATOTAL, que sirve para resumir la información de facturación en una lista que puede verse por pantalla o enviarse a una impresora o a un archivo.

Para proceder a ejecutar la utilidad, nos situaremos en el directorio SYS:SYSTEM, o en su defecto también podemos establecer una unidad de búsqueda hacia él, y a continuación escribiremos la orden siguiente:

ATOTAL

y para redirigir el listado a un archivo determinado:

ATOTAL > *nombre_de_archivo*

Funciones Hash. MD-5.

José Antonio Mendoza

Mediante las funciones hash tratamos de resolver el problema de la autenticidad de un documento en formato digital, al igual que en el mundo de los documentos clásicos, es decir, los de tinta y papel. La firma y la caligrafía identifican a una persona de forma unívoca. En el mundo judicial tenemos como muestra a los peritos caligráficos, que son los que dan fe en un juicio sobre la autenticidad de una firma o un manuscrito.

Cuando un documento tradicional va firmado por su autor, se garantiza :

1. El receptor puede verificar que el documento ha sido enviado por el autor.
2. El autor no puede negar que el documento no haya sido realizado por él.
3. El receptor no puede alterar el contenido del documento.

En la televisión siempre vemos que, cuando se firma un acuerdo, las dos partes no paran de firmar folios y folios y un señor se dedica a secar la tinta de las firmas para que ésta no se emborrone al ponerse en contacto con la superficie de otro folio. Todos los folios de un contrato son numerados y firmados por ambas partes. Al concluir el acto, una parte entrega a la otra un ejemplar del contrato y después viene el apretón de manos, los abrazos, quizás una copita de cava y unos aplausos si hay público presente. Tras este breve paréntesis de protocolo de firmas de contratos adentrémonos en el mundo de las firmas digitales y la forma en que solucionan el problema.

La autenticidad de un mensaje se puede conseguir de muchas formas: mediante criptografía, ya sea pública o clásica, redundancia cifrada y mediante funciones hash, objeto de nuestro artículo. Los métodos criptográficos plantean problemas con el intercambio de claves y su seguridad. Uno puede decir que no es el autor de un determinado documento cifrado con su clave privada, argumentando que se la han robado o la ha perdido, y que un supuesto falsificador la ha empleado para generar el mensaje del que se quiere negar la autoría. También es posible que no sea necesaria la privacidad de un documento, sólo garantizar su integridad y autoría. Veamos pues en qué consisten las funciones hash.

Funciones hash

Las funciones hash son el primer paso para la generación de la firma digital de un documento. Estas funciones nos dan la huella de un mensaje, esto es, un conjunto de bits que se calculan mediante la función hash partiendo del mensaje. La firma digital del mensaje es algo más complejo que incluye el cifrado de la huella, los protocolos de reparto de claves, la fecha y la

En el mundo de la seguridad informática no todo consiste en cifrar mensajes, también es preciso poder autentificarlos. Al igual que los documentos impresos se autentifican con la firma manuscrita de una persona, los documentos en formato digital necesitan métodos que realicen la misma función.

hora en la que se genera el mensaje, los árbitros que actúan como entidad de confianza entre los dos interlocutores de una comunicación y un largo etcétera.

Una función cualquiera no puede ser función hash ya que debe poseer una serie de propiedades sin las cuales el valor generado no serviría como huella. Llamaremos "h" al valor huella, "H" a la función huella y "M" al mensaje del que queremos calcular su huella, por tanto h=H(M). Las propiedades de H más importantes son :

1. El valor resultante h tiene una longitud fija.

2. El mensaje M puede tener cualquier longitud.

3. Dada una huella "s", es computacionalmente imposible hallar un mensaje que la genere con H.
   Es decir s=H(R). R es computacionalmente imposible de encontrar.

4.- Es computacionalmente imposible encontrar dos mensajes distintos que generen la misma huella.

Mediante la firma digital se pretende que un mensaje no se pueda alterar y que tenga autor

Recordemos que "computacionalmente imposible hallar" no quiere decir que no exista forma de hallar sino que con los ordenadores actuales nos llevaría muchísimo tiempo realizar los cálculos. Podemos observar que la alteración de un mensaje tendría como resultado que ya no casara el mensaje alterado con la huella asociada al mensaje inicial, garantizando así su integridad. La manera habitual en que una función hash genera la huella suele ser dividiendo el mensaje en bloques de n bits. Si el último bloque no llega a ser de n bits se completa con ceros o unos o un valor predeterminado, igual en el emisor y el receptor. La primera función hash que se nos puede ocurrir pudiera ser la suma OR-Exclusivo de los bits i-ésimos de cada bloque. Pero esta solución sólo valdría para mensajes completamente aleatorios. En cuanto tengamos texto, voz, imágenes, en suma, datos con información, las huellas así generadas no serían equiprobables, aumentando la probabilidad de que dos mensajes distintos den la misma huella.

Para evitar esta circunstancia se recurre a barajar los bits lo máximo posible, para conseguir que las huellas generadas sean equiprobables, es decir que dado un mensaje su huella no sea parecida a la de un mensaje parecido.

Existen multitud de funciones hash, cada una con unas determinadas propiedades de robustez, rapidez de codificación, decodificación, etc. Podemos destacar a *snefru* desarrollada por Ralph Merkle, el mismo del algoritmo de la mochila; MD-5 desarrollada por Ron Rivest, el mismo que desarrolló el algoritmo de clave pública del RSA; SHA (*Secure Hash Algorithm*) desarrollada por la NSA (*National Security Agency*) y adoptada por la NIST (*National Institute of Standard and Technology*).

Antes de ver en qué consiste la función hash MD-5, empleada por el programa PGP (*Pretty Good Privacy*) desarrollado por Phil Zimmermann, veremos uno de los posibles ataques que pueden sufrir dichas funciones.

Ataque del cumpleaños

Supongamos una secretaria que quiere engañar a su jefe haciendo pasar mensajes escritos por ella pero firmados por su jefe. Antes bastaba con que el jefe firmara el documento en el sitio adecuado y, lógicamente, que no leyera nada de lo que la secretaria le daba a firmar, pero ahora las cosas han cambiado y los jefes piden un fichero con el texto a enviar. ¿Cómo procedería nuestra secretaria ? La secretaria podría preparar dos cartas, una la que quiere enviar el jefe y otra la que ella quiere atribuir al jefe. Pero estas cartas son un poco especiales. Ambas tienen m parejas de palabras o expresiones sinónimas, por ejemplo: Ayer { Francisco / Paco } le propuso a { José / Pepe } comprar un { Automóvil / coche } de potentes { faros / luces } ..., y así hasta m parejas. A primera vista vemos que calculando todas las huellas posibles, 2^m, de uno y otro mensaje con seguridad encontraremos dos que nos den la misma.

Pero, ¿de quién es el cumpleaños? Este ataque está basado en la paradoja del cumpleaños. La teoría de la probabilidad muchas veces nos conduce a resultados que a primera vista podría parecer que hubieran sido otros completamente distintos. Si nos preguntáramos por el mínimo número de personas que en un aula tiene que haber para que la probabilidad de que dos de ellas cumplan los años el mismo día fuera mayor o igual a un medio, intuitivamente pensaríamos que con cien personas ya se podría dar una coincidencia. Sin embargo, la teoría de la probabilidad nos dice que dicho número es tan solo de veintitrés. Llevado esto a nuestro caso significaría que la cantidad de huellas a calcular no es 2^m sino $2*2^{m/2}$, las $2^{m/2}$ del mensaje verdadero y las $2^{m/2}$ del mensaje falso, lo cual reduce substancialmente el número de huellas a calcular. Veamos las probabilidades de la paradoja del cumpleaños.

Para hallar la expresión de dicha probabilidad, tenemos que hacer los siguientes cálculos :

- Sea P(365,k) la probabilidad de que con k elementos al menos dos de ellos coincidan en su valor

- Sea S(365,k) la probabilidad del suceso complementario, es decir, que en los k elementos no coincida ninguno.

AWS

C/ SERRANO JOVER, 3
TELF.: 559 82 86
METRO ARGÜELLES

C/ GINZO DE LIMIA, 48
TELF.: 559 82 86
METRO BARRIO DEL PILAR

Acer

ACER NOTE 350

PENTIUM 120
8 MB EDO RAM
HD 1 GIGA
PANTALLA 10.4" DSTN

184.900

AWS LITE

PENTIUM 133
PLACA OPTI 256 CACHÉ
8 MB EDO RAM
DISCO DURO 1 GIGA
MONITOR SVGA 14" 1024 N.E.
CD ROM x 16
T. SONIDO COMPATIBLE ALTAVOCES

intel inside pentium

98.900

TEXAS INSTRUMENTS

EXTENSA 610

PENTIUM 120
8 MB EDO RAM
HD 1 GIGA
PANTALLA 11.3" DSTN

202.000

TOSHIBA

SATELLITE 220 CS

PENTIUM 133
16 MB EDO RAM
HD 1.3 GIGAS
PANTALLA 12.1" DSTN
CD ROM x 10

289.900

AWS MAX

PENTIUM 233 MMX
PLACA TX 512 CACHÉ
64 MB EDO RAM
DISCO DURO 3.2 GIGAS
VGA MATROX MYSTIQUE 2 MB
MONITOR SVGA 17" 1280 DIGITAL
CD ROM x 24
SOUND BLASTER 64 GOLD
ALTAVOCES 240 W, MICRÓFONO

MMX intel inside pentium

CON 5 AÑOS DE GARANTÍA

289.900

TOSHIBA

TECRA 520 CDT

PENTIUM 166 MMX
32 MB EDO RAM
HD 2 GIGAS
PANTALLA 12.1" TFT
CD ROM x 10

710.000

OFERTA DE AMPLIACION DE 486 A PENTIUM

MICRO INTEL PENTIUM 120
PLACA OPTI 256 CACHÉ
CAMBIO DE RAM A 72 C (SI PROCEDE)
INSTALACIÓN Y MANO DE OBRA

intel inside pentium

25.000

TARJETAS PCMCIA

FAX-MODEM 33.600 .... 20.900
TARJETA DE RED 14.400
FAX-MODEM + T. RED ... 41.600

SOMOS ESPECIALISTAS EN PORTÁTILES

I.V.A. no incluido. Ofertas válidas salvo error tipográfico, hasta fin de existencias.

¿TE ATREVES?

Desde **Sólo Programadores** estamos preparándonos para un torneo muy especial, de ordenador a ordenador, de programador a programador. El torneo de ajedrez está empezando a tomar forma y éstas son las bases, aunque también podéis informaros en AWS en la calle Serrano Jover, 3, de Madrid, o en el teléfono (91) 5 42 50 87. Para conocer cuál será el alcance de la participación, os pedimos que os inscribáis enviando vuestros datos a la dirección de AWS o a la dirección de e-mail: **solop@towercom.es**. El periodo de inscripción ya ha comenzado y terminará el 30 de noviembre, y la fecha límite para la recepción de los programas es el 15 de diciembre de 1997. La celebración del torneo será anunciada con suficiente antelación y podréis participar en persona o desde vuestra propia ciudad, ya que vuestro programa os representará y en todo momento estará presente un notario.

BASES DEL TORNEO

1. Se presentará un programa que juegue al ajedrez recibiendo la jugada del contrario y devolviendo la propia. El tiempo de la partida será medido por un juez y la partida se jugará físicamente en una mesa con un tablero siguiendo las instrucciones que cada ordenador dé; cada ordenador avisará de que ha elegido un movimiento con un pitido y mostrará su jugada en la pantalla de la forma: casilla a mover - casilla destino. La notación de la casilla vendrá dada por su columna y su fila. Para las columnas se utilizarán las de la A a la H siendo la columna A la primera columna de la izquierda, visto el tablero desde la posición de las blancas. Para las filas utilizaremos los números del 1 al 8 siendo la fila 1 la más cercana a nosotros (visto el tablero desde el lado de las blancas). Como ejemplo, el movimiento del peón de rey dos lugares hacia adelante sería: "E2 - E4". La jugada que hace el contrario se anotará en el ordenador de forma manual tecleando "E2-E4" <RETURN>.
2. El programa jugará al ajedrez según las normas de este juego, admitiendo todos sus movimientos.
3. El programa se ejecutará bajo Windows 95, MS Dos 6.22 y Linux, en un Pentium 166 con 16 MB de RAM.
4. El programa podrá jugar con las blancas o con las negras indistintamente.
5. Se debe entregar el código fuente, la versión del compilador utilizado y una breve documentación para su utilización.
6. Un programa podrá ser descalificado si realiza un movimiento no válido y esto deberá ser advertido por el contrario; por lo tanto, un movimiento será válido siempre que el otro programa no nos advierta de lo contrario. De la misma forma, si un programa nos advierte de un movimiento erróneo y éste no lo es, será descalificado.
7. Los jueces son los encargados de advertir las posiciones de tablas, el programa no advertirá de ello.
8. El programa debe detectar si ha ganado la partida y quedará descalificado ante cualquier detección errónea. No se avisará al contrario en el caso de que deje al rey al descubierto y ganará la partida el que antes mate al rey.
9. La metodología del concurso se reservará hasta que sea conocido el número de participantes (eliminatoria, puntos, etc.)
10. No se podrán utilizar en el programa bases de datos adicionales.
11. Los derechos del programa ganador pasarán a la propiedad de la empresa Tower Communications, SRL.
12. El premio consistirá en un Pentium PRO 200 con 64 MB RAM, VIRGE4 MB, CDX 16, disco duro 5 GB, monitor 17", teclado, ratón y disquetera.

CONCURSO DE PROGRAMACIÓN

SÓLO PROGRAMADORES Y AWS INFORMATICA
SERRANO JOVER 3, MADRID

MÁS DATOS EN EL PRÓXIMO NÚMERO

Como son sucesos complementarios $P(365,k) + S(365,k) = 1$, sucede uno o sucede el otro. O sale cara o sale cruz.

Sea N el número de veces que podemos tener k valores sin que se repita ninguno.

$N=365*364*363*...*(365-k+1)= [(365)\ ! / (365-k)\ !]$

El primer elemento podrá tomar cualquier valor, el segundo uno menos, el tercero dos menos y así sucesivamente, para que no haya repeticiones. Si eliminamos la restricción de que no se pueda repetir ningún elemento tendremos :

$N=365*365*365*...*365= 365^k$

Por lo tanto, la probabilidad de que no se repita ningún valor en k elementos será casos favorables dividido por casos posibles. Es decir:

$S(365,k)=365\ !/((365-k)\ !365^k)$ y $P(365,k)=1- [365\ !/((365-k)\ !365^k)]$

Igualando la expresión anterior a 0.5 y despejando sale k=23. A primera vista parece imposible despejar, ni siquiera calcular nada en la expresión anterior con factorial de 365, el truco está en aproximar el cociente por $e^{-[(kx(k-1))/(2*365)]}$. Despejando k y aproximando llegamos a $k=1.17*365^{1/2}=23$

Las funciones HASH nos proporcionan la huella de un mensaje

Simplemente con veintitrés personas, la probabilidad de que dos de ellas hayan nacido el mismo día es de un medio, lo cual resulta, cuando menos, paradójico. Para m grande $k=m^{1/2}$. Con todo esto podemos llegar a deducir que para que nuestra función hash tenga sentido y no sea vulnerable a dicho ataque, la huella debe tener una cantidad importante de bits.

El algoritmo MD5

MD5 es la función hash de PGP (*Pretty Good Privacy*). PGP hace uso de muchos algoritmos, IDEA como cifrador de mensajes, RSA para encriptar la huella del mensaje y la clave de sesión, ZIP para comprimir los datos y por último el algoritmo que vamos a ver, MD5, para calcular la huella del mensaje.

Para poner en claro la utilización de todos estos algoritmos veamos el proceso secuencialmente, si lo que queremos conseguir es confidencialidad y autentificación :

1. Se escribe el mensaje a transmitir.
2. Se genera la huella del mensaje con MD5.
3. La huella se encripta con el RSA, utilizando como clave la privada del emisor.
4. Se concatena el mensaje y la huella cifrada.
5. Se comprime con ZIP.
6. Se cifra el resultado con idea utilizando una clave de sesión.
7. La clave de sesión se cifra con RSA utilizando como clave la pública del receptor.
8. Se concatena el texto cifrado del mensaje y la clave de sesión también cifrada.
9. Se manda al canal.

El algoritmo está dividido en cinco pasos bien diferenciados

10. En recepción se realizan las operaciones inversas y, además, en el orden contrario. La clave de sesión será la misma, pero lo que antes se cifró con la pública ahora se descifra con la privada, y lo que se cifró con la privada ahora se descifrará con la pública.

MD5 es un algoritmo que tiene como entrada un mensaje de cualquier longitud y como salida la huella de dicho mensaje con una longitud de 128 bits. Internamente MD5 procesa el mensaje en bloques de 512 bits, realizando con ellos las operaciones que ahora veremos. El algoritmo está dividido en cinco pasos bien diferenciados que procedemos a describir a continuación :

• Primer paso: Añadir bits de relleno.

Si el algoritmo procesa los bits en bloques de 512 y nuestro mensaje puede tener una longitud arbitraria, nos vemos en la obligación de añadir bits hasta conseguir que la longitud sea múltiplo de 512 bits. En este paso sólo añadiremos los bits necesarios para conseguir que la longitud sea igual a $L=448 \bmod 512$. En otras palabras, añadiremos bits hasta conseguir que a la nueva longitud le falten 64 bits para ser múltiplo de 512. ¿Cómo son los bits que le añadimos ?, pues un uno seguido de los ceros necesarios.

• Segundo paso: Añadir la longitud.

Antes no hemos añadido 64 bits para conseguir que la nueva longitud fuera múltiplo de 512, sino que lo hemos hecho con la intención de reservar 64 bits para añadir al mensaje la longitud inicial que tenía. En caso de que la longitud del mensaje exceda a 2^{64} se añadirían los sesenta y cuatro bits de menos peso de la longitud. Añadidos estos bits, nuestro nuevo mensaje tiene en este momento una longitud que es múltiplo de 512 bits, garantizando de esta forma que el mensaje pueda ser procesado en bloques de 512 bits.

- Tercer paso: Inicialización del MD.

El algoritmo necesita un búfer de 128 bits donde guardar valores intermedios y entregar el resultado. Pero para que el algoritmo empiece a computar necesita que dicho búfer se encuentre inicializado con un determinado valor. El búfer de 128 bits está dividido en cuatro registros de 32 bits cada uno: A, B, C, D. Los valores iniciales son:

A = 01234567
B = 89ABCDEF
C = FEDCBA98
D = 76543210

- Cuarto paso: Proceso del mensaje en bloques de 512 bits.

Este es el paso fundamental del algoritmo, donde se realizan la totalidad de las operaciones.

El algoritmo de este paso se ejecutará tantas veces como bloques de 512 bits tenga el mensaje. Tendrá como entradas, por un lado, los 512 bits del bloque a procesar y por el otro el valor presente en el buffer MD, compuesto por los cuatro registros: A, B, C, D. La salida la pondrá en el buffer MD actualizando así su valor. Tras la última ejecución del algoritmo, el valor presente en dicho buffer contendrá el resultado, es decir, la huella de nuestro mensaje.

MD5 es la función HASH que se utiliza en el PGP

El algoritmo se divide en cuatro bucles. La estructura de los bucles es idéntica, pero difieren en una función, que tomará una expresión diferente para cada uno de ellos. A su vez la función se ejecutará dieciséis veces, cada una afectará a diferentes variables. Lo vemos mejor con un ejemplo:

La función del primero de los cuatro bucles es a = b + ((a + F (b, c, d) + X[k] + T[i]) <<< s), y su representación: [abcd k s i]; y la primera ejecución del primer bucle [ABCD 0 7 1], la segunda [DABC 1 12 2], es decir en la expresión de la fórmula sustituimos las minúsculas por las mayúsculas y números. En el segundo bucle tendremos la misma función salvo que en vez de F(b,c,d) será G(b,c,d) el tercero H(b,c,d) y el cuarto I(b,c,d). Dichas funciones son las siguientes:

$F(b,c,d) = (b \cdot c) \vee (\bar{b} \cdot d)$
$G(b,c,d) = (b \cdot d) \vee (c \cdot \bar{d})$
$H(b,c,d) = b \barwedge c \barwedge d$
$I(b,c,d) = c \barwedge (b \cdot \bar{d})$

MD5 tiene como entrada un mensaje de cualquier longitud y como salida la huella de dicho mensaje con una longitud de 128 bits

Vistas las funciones, queda por ver la X[k] y la T[i]. La primera es el k-ésimo grupo de 32 bits de los 512 del bloque con que se ejecuta cada paso, obviamente 512=16*32. La T[i] es un poco más complicada, trata de ser una tabla de valores aleatorios donde la i varía de 1 a 64. Cada uno de los bucles tiene dieciséis valores de T[i]. El primero de T[1] a T[16] el segundo de T[17] a T[32] y así los demás hasta T[64]. Estos valores se generan multiplicando por 4294967296 el valor absoluto del sen(i), i=1,2,3,4,...,64 en radianes, y tomando la parte entera. Sólo queda explicar el significado de (w) <<< s, que indica la rotación circular a la izquierda de s posiciones de w.

En las expresiones de F,G,H e I, el punto significa la función AND, la $\vee$ la función OR, la barrita encima de la letra la operación NOT y el $\barwedge$ la operación XOR.

- Qùinto paso: Salida.

Después de haber procesado todos los bloques del mensaje tendremos la huella en el buffer MD compuesto por los registros A, B, C, D.

Para ver mejor en qué consiste el paso cuarto, que es el más complejo, podemos ver el pseudocódigo importado de la RFC1321. Las RFC son documentos a disposición del público en la internet donde vienen definidos los estándares relacionados con ella. Por ejemplo el TCP/IP, Kerberos, SNMP, X.25, UDP y un largo etcétera.

```
/* Process each 16-word block. */
  For i = 0 to N/16-1 do
    /* Copy block i into X. */
    For j = 0 to 15 do
      Set X[j] to M[i*16+j].
    end /* of loop on j */

    /* Save A as AA, B as BB, C as CC, and D as
       DD. */

    AA = A
    BB = B
    CC = C
    DD = D

    /* Round 1. */
    /* Let [abcd k s i] denote the operation
       a = b + ((a + F(b,c,d) + X[k] + T[i])
                        <<< s). */

    /* Do the following 16 operations. */
       [ABCD  0  7  1]
       [DABC  1 12  2]
       [CDAB  2 17  3]
       [BCDA  3 22  4]
       [ABCD  4  7  5]
       [DABC  5 12  6]
       [CDAB  6 17  7]
       [BCDA  7 22  8]
       [ABCD  8  7  9]
       [DABC  9 12 10]
       [CDAB 10 17 11]
       [BCDA 11 22 12]
```

```
    [ABCD 12  7 13]
    [DABC 13 12 14]
    [CDAB 14 17 15]
    [BCDA 15 22 16]

/* Round 2. */
/* Let [abcd k s i] denote the operation
     a = b + ((a + G(b,c,d) + X[k] + T[i])
     <<< s). */

 /* Do the following 16 operations. */
    [ABCD  1  5 17]
    [DABC  6  9 18]
    [CDAB 11 14 19]
    [BCDA  0 20 20]
    [ABCD  5  5 21]
    [DABC 10  9 22]
    [CDAB 15 14 23]
    [BCDA  4 20 24]
    [ABCD  9  5 25]
    [DABC 14  9 26]
    [CDAB  3 14 27]
    [BCDA  8 20 28]
    [ABCD 13  5 29]
    [DABC  2  9 30]
    [CDAB  7 14 31]
    [BCDA 12 20 32]

/* Round 3. */
/* Let [abcd k s t] denote the operation
     a = b + ((a + H(b,c,d) + X[k] + T[i])
     <<< s). */

/* Do the following 16 operations. */
    [ABCD  5  4 33]
    [DABC  8 11 34]
    [CDAB 11 16 35]
    [BCDA 14 23 36]
    [ABCD  1  4 37]
    [DABC  4 11 38]
    [CDAB  7 16 39]
    [BCDA 10 23 40]
    [ABCD 13  4 41]
    [DABC  0 11 42]
    [CDAB  3 16 43]
    [BCDA  6 23 44]
    [ABCD  9  4 45]
    [DABC 12 11 46]
    [CDAB 15 16 47]
    [BCDA  2 23 48]

/* Round 4. */
/* Let [abcd k s t] denote the operation
     a = b + ((a + I(b,c,d) + X[k] + T[i])
                 <<< s). */

/* Do the following 16 operations. */
    [ABCD  0  6 49]
    [DABC  7 10 50]
    [CDAB 14 15 51]
    [BCDA  5 21 52]
    [ABCD 12  6 53]
    [DABC  3 10 54]
    [CDAB 10 15 55]
    [BCDA  1 21 56]
    [ABCD  8  6 57]
    [DABC 15 10 58]
    [CDAB  6 15 59]
    [BCDA 13 21 60]
    [ABCD  4  6 61]
    [DABC 11 10 62]
    [CDAB  2 15 63]
    [BCDA  9 21 64]

 /* Then perform the following additions.
     (That is increment each
    of the four registers by the value it had before
     this block
    was started.) */

  A = A + AA
  B = B + BB
  C = C + CC
  D = D + DD
 end
  /* of loop on i */
```

Los valores de las T[i] son los siguientes :

Primer grupo :

```
T[1] = D76AA478     T[2]= E8C7B756
T[3] =  242070DB    T[4] = C1BDCEEE
T[5] = F57C0FAF     T[6] = 4787C62A
T[7]= A8304613      T[8] =  FD469501
T[9] = 698098D8     T[10] =  8B44F7AF
T[11] = FFFF5BB1    T[12] = 895CD7BE
T[13] = 6B901122    T[14] = FD987193
T[15]= A679438E     T[16] = 49B40821.
```

Segundo grupo :

```
T[17] = F61E2562    T[18] = C040B340
T[19] = 265E5A51    T[20] = E9B6C7AA
T[21] = D62F105D    T[22] = 2441453
T[23] = D8A1E681    T[24] = E7D3FBC8
T[25] = 21E1CDE6    T[26] = C33707D6
T[27] = F4D50D87    T[28] = 455A14ED
T[29] = A9E3E905    T[30] = FCEFA3F8
T[31] = 676F02D9    T[32] = 8D2A4C8A
```

Tercer grupo :

```
T[33] = FFFA3942    T[34] =8771F681
T[35] = 6D9D6122    T[36] = FDE5380C
T[37] = A4BEEA44    T[38] = 4BDECFA9
T[39] = F6BB4B60    T[40] = BEBFBC70
T[41] =  289B7EC6   T[42] = EAA127FA
T[43] = D4EF3085    T[44] = 4881D05
T[45] = D9D4D039    T[46] =  E6DB99E5
T[47] = 1FA27CF8    T[48] = C4AC5665
```

Cuarto grupo :

```
T[49] = F4292244    T[50] = 432AFF97
T[51] = AB9423A7    T[52] = FC93A039
T[53] = 655B59C3    T[54] = 8F0CCC92
T[55] = FFEFF47D    T[56] = 85845DD1
T[57] = 6FA87E4F    T[58] = FE2CE6E0
T[59] = A3014314    T[60] =  4E0811A1
T[61] = F7537E82    T[62] = BD3AF235
T[63] = 2AD7D2BB    T[64] = EB86D391
```

Conclusiones

La fuerza de una función hash se encuentra por tanto en dos factores, la longitud de la huella que es capaz de generar y la capacidad que tenga para hacerla lo más aleatoria posible, sin por ello, hacerla independiente del mensaje. En el caso particular de la MD5, la huella es de 128 bits, lo que hace que un ataque del cumpleaños sea impracticable. A su vez, la estructura interna del algoritmo, muy variante en la expresión de las funciones y la inclusión de números prácticamente aleatorios como son los de las T[i], hacen muy robusto al ataque diferencial.

Bibliografía

- *Network and Internetwork Security*, Autor William Stallings, editorial Prentice-Hall.
- *Computer Networks*, Autor Andrew S. Tanenbaum, editorial Prentice-Hall.
- RFC1321, Autor R. Rivest.

HERRAMIENTAS DE PROGRAMACIÓN

DirectX y DirectX SDK 3.0a

Jorge Figueroa

Las librerías DirectX, debido a que han sido diseñadas por la propia Microsoft, se convertirán en unas extensiones estándar del propio sistema operativo Windows.
En este artículo desvelamos todos sus secretos.

Las APIs que conforman la DirectX se convertirán en estándar en un futuro no muy lejano, ya que, además de lo práctico que resulta para los desarrolladores utilizarlas en sus aplicaciones, en lo que a tiempo de programación ahorrado se refiere, van a ser incluidas casi seguro "de fábrica" con las próximas versiones de Windows.

DirectX es el nombre genérico de un conjunto de APIs para Windows

Como complemento a esta serie de nuevas APIs, Microsoft ha desarrollado un Kit para desarrolladores (principalmente de videojuegos) que hace que sea más fácil programar aplicaciones que hagan provecho de ellas. Recientemente se acaba de presentar la versión 3.0 de este Kit de desarrollo que se llama DirectX SDK y que incluye también la última versión de las librerías DirectX, las cuales aportan muchas mejoras en general respecto a las anteriores versiones.

En el presente artículo se va a detallar todo lo que le aportan estas librerías al programador que se decida a usarlas en sus programas así como una valoración general del propio kit de desarrollo.

¿Qué es la DIRECTX?

El Sistema Operativo Windows, desde siempre, se ha caracterizado por su lentitud en todo lo referente al acceso de pantalla y hardware en general y esto lo convierte en un pobre candidato para la realización de aplicaciones basadas en él que hagan uso intensivo de todos los recursos del sistema, como son juegos y programas interactivos en general, puesto que estas aplicaciones destacan precisamente por el hecho de que hacen un uso exagerado de gráficos 2D y 3D, sonido y efectos digitales, entre otras cosas.

Dicho esto, y dado que el futuro de toda la informática pasa en todos sus aspectos por la multimedia y la realidad virtual, Microsoft ha creado un conjunto de librerías estándar que se encuentran englobadas bajo el nombre común de DirectX y que se instalan en el sistema operativo como una extensión del mismo proporcionándole, una vez situadas en el núcleo de Windows, la posibilidad de que los programas diseñados para aprovechar esta tecnología dispongan de todas las funciones más comunes que usa cualquier programador en las aplicaciones relacionadas con entrada y salida de datos multimedia.

Con todo lo dicho, se consiguen principalmente dos ventajas:

1- Los diseñadores, por un lado, no tienen que preocuparse de crear todas las rutinas gráficas 2D/ 3D, ni de sonido, ni nada parecido, puesto que todo ello está ya disponible en los APIs del S.O. y por lo tanto, pueden dedicarse más al proyecto en sí mismo.

2- Por otro lado, el código de las aplicaciones ocupan menos espacio en disco, ya que se comparten más recursos del sistema y las aplicaciones se convierten casi en simples cúmulos de llamadas a funciones externas con un montón de gráficos, textos y sonido por gestionar.

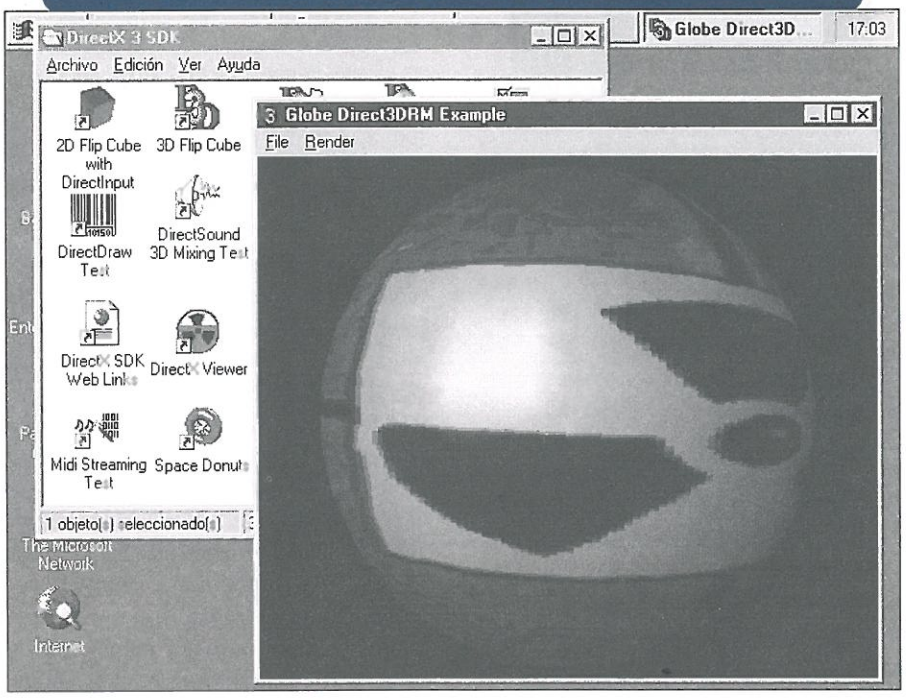
Direct Draw permite generar y manipular gráficos 2D a mayor velocidad

Por otro lado, no todo es positivo y tenemos también que la librería DIRECTX no es todo lo rápida ni eficaz que sería deseable para muchos programadores, con lo cual, las posturas de opinión actuales respecto a ella y su utilización son muy diversas.

Es casi seguro que en un futuro más o menos próximo, la DirectX se convertirá en un estándar que formará parte de Windows como un componente más del sistema (según ha declarado la propia Microsoft) y que el uso de dichas librerías se hará tan común como el uso que hacen las aplicaciones actuales de las propias funciones de la interfaz gráfica de Windows. En general, podemos decir que su éxito dependerá de cómo evolucionen estas APIs en las próximas versiones y de cómo vaya evolucionando la aceptación hacia las mismas por parte de los diversos desarrolladores.

Mientras se sigue discutiendo sobre el futuro de las DirectX, en el presente artículo vamos a desvelar todas las posibilidades de que dispone el programador que se decide a usarlas en sus propias aplicaciones, detallando tanto la propia API como el Kit de desarrollo que Microsoft ha creado como complemento para los interesados en su uso. Con todo ello, el propio lector podrá hacerse una opinión propia.

Ejecución de renderización de un Ejemplo 3D del Kit en una ventana.



Qué se incluye bajo el nombre DirectX

Como hemos dicho, bajo el nombre de DirectX están incluidas muchas funciones que se encuentran divididas por áreas de programación (utilidad). Así pues, disponemos de las siguientes APIs:

- *Direct Draw*: Funciones relacionadas con gráficos 2D.
- *Direct 3D:* Todo lo relacionado con las 3D.
- *Direct Input:* Funciones para lectura de datos provenientes de periféricos de control (teclado, mouse, etc.).
- *Direct Sound:* Funciones de sonido.
- *Direct Play:* Funciones de conectividad.
- *Direct Setup:* Funciones para manejo de instalaciones.

Direct 3D es un completo engine 3D con el que podemos crear imágenes 3D

A continuación vamos a detallar las posibilidades y funciones que nos ofrecen las principales APIs mencionadas para que el programador pueda saber si le conviene o no aprender a usarlas.

Direct Draw

El API Direct Draw está creado para que el programador pueda acceder a la memoria de vídeo directamente desde

Tabla 1.
Algunas funciones que aparecen en DIRECT DRAW.

IDirectDraw::QueryInterface
IDirectDraw::RestoreDisplayMode
IDirectDraw::SetCooperativeLevel
IDirectDraw::SetDisplayMode
IDirectDrawSurface2::Blt
IDirectDrawSurface2::BltBatch
IDirectDrawSurface2::UpdateOverlay
IDirectDraw2::EnumDisplayModes
IDirectDraw2::GetAvailableVidMem
IDirectDraw2::SetDisplayMode
IDirectDrawPalette::SetEntries
IDirectDrawClipper::Initialize
IDirectDrawSurface2::BltFast
IDirectDrawSurface2::GetDDInterface
IDirectDrawSurface2::PageLock
IDirectDrawSurface2::PageUnlock.

sus programas, realizar *flips* de búferes de pantalla situados en memoria de vídeo para acelerar Scrolls, usar Z-Buffer por hardware, poder generar efectos de escalado de bitmaps por hardware, acceder a áreas especiales de memoria y proporcionar soporte para transparencias hardware entre otras cosas.

Al ser todas estas funciones de un nivel más bajo que las funciones que usa un programa Windows clásico, se consigue minimizar los tiempos necesarios para la manipulación de la memoria gráfica y aprovechar las capacidades hardware gráficas 2D (aunque también alguna 3D). En la tabla 1 observamos ejemplos de funciones que hay dentro de esta API donde podemos observar lo fácil que es usar y entender dichas funciones.

Como se puede comprobar, todas las llamadas se realizan mediante los conocidos como COM (*Component Object Model*) de C++.

De esta forma las funciones quedan más organizadas y el código de aplicación se ve más claro.

Direct 3D

Podemos decir que el API destinado a generación de gráficos tridimensionales es el más importante de todo el conjunto, puesto que la aceptación que va a tener todo el DirectX depende en gran medida de los resultados que se puedan obtener con él.

La causa es simple: la tercera dimensión es el futuro de todas las áreas de interfaz con el usuario, incluyendo la simulación de mundos 3D por internet, la realidad virtual con cascos, etc. La utilidad de disponer de una API para generación de graficos 3D en un sistema como Windows, es que los desarrolladores se ahorran tener que realizar código especial para cada uno de los más importantes modelos de tarjeta aceleradora 3D del mercado, ya que en este sistema operativo el fabricante es el que suministra los drivers de interfaz y los programadores sólo tienen que usar las funciones de la API, que son siempre las mismas. Con ello, tenemos que todas las operaciones gráficas 3D las podemos realizar por emulación software o directamente por hardware si los recursos estan disponibles, dependiendo de si la tarjeta instalada soporta o no dichas capacidades (al más puro estilo de cómo soportan el coprocesador matemático los programas de C).

Hay que recordar que las capacidades hardware qure podemos encontrarnos en una tarjeta 3D actual pueden ser por ejemplo: z-buffering, antialiasing, alpha blending, efectos atmosféricos mipmapping, corrección de perspectiva en texturas, etc. y que se caracterizan todas ellas por su alto coste en cuanto a tiempo de procesado en caso de tener que realizarse por software.

Si examinamos todo el API, descubriremos que las posibilidades operativas son muy amplias. Así tenemos que podemos realizar rellenados de polígonos en modo flat, gouraud, Phong, uso de texturas, uso de luces de varios tipos, como las especulares, manejo de paletas, fog, correción de perspectiva de varios tipos y, en resumen, todo lo que podemos esperar de un moderno engine 3D. Pese a todo lo dicho, también es cierto que muchos programadores actuales de aplicaciones 3D no están demasiado contentos con el rendimiento de esta API y se supone que su aceptación irá aumentando a medida que

El programa 3D test funcionando.

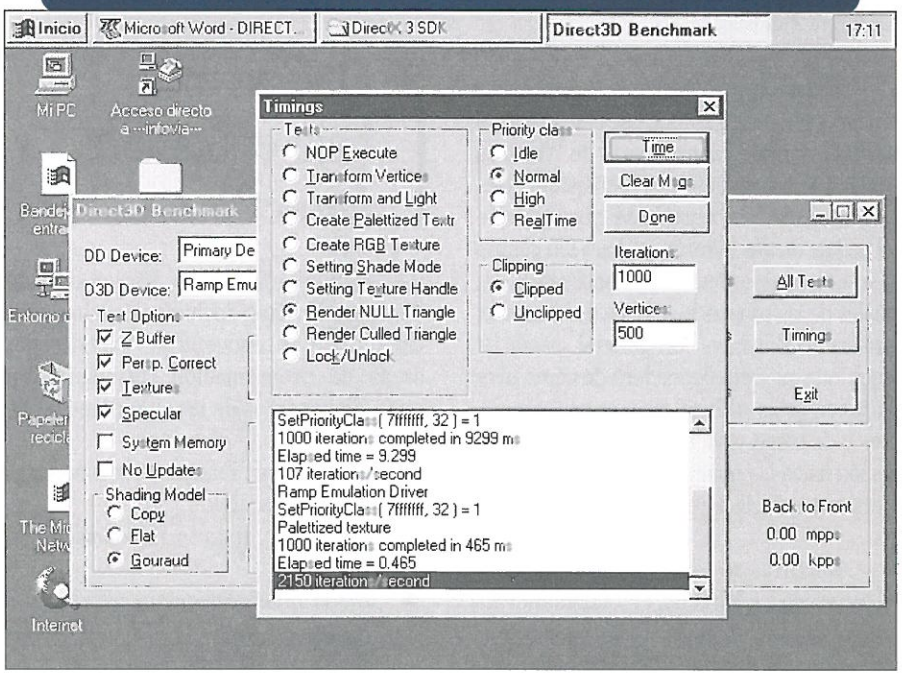


Tabla 2. Ejemplo de una rutina que usa Direct 3D para limpiar el búfer de vídeo, renderizar un frame pendiente y actualizar la pantalla.

```
static BOOL
BUCLE_Renderizado()
{
 HRESULT rval;

 // Alamcena la escena 3D.
 rval = myglobs.scene->lpVtbl->Move(myglobs.scene, D3DVAL(1.0));
 if (rval != D3DRM_OK) {
  return FALSE;
 }

 // Limpia el búffer de imagen:
 rval = myglobs.view->lpVtbl->Clear(myglobs.view);
 if (rval != D3DRM_OK) {
  return FALSE;
 }

 // Renderiza la escena en búffer
 rval = myglobs.view->lpVtbl->Render(myglobs.view, myglobs.scene);
 if (rval != D3DRM_OK) {
  return FALSE;
 }

 // Actualiza la pantalla
 rval = myglobs.dev->lpVtbl->Update(myglobs.dev);
 if (rval != D3DRM_OK) {
  return FALSE;
 }
 return TRUE;
}
```

vayan saliendo versiones más modernas y optimizadas del Direct 3D.

Un dato que puede interesar es que se puede usar el Direct 3D de dos formas diferentes: los llamados modos *Retained* y *Inmediate*. El modo **Retained** es un API de alto nivel que está diseñado especialmente para la manipulación de objetos y de imágenes 3D, lo cual permite a aplicaciones ya existentes, y también a otras por hacer, beneficiarse de la generación de escenas 3D de una forma fácil, desde el punto de vista del programador, debido fundamentalmente a que acepta el uso de key frames, estructuras de objetos y demás conceptos similares.

Direct Input permite controlar cualquier tipo de teclado y apuntador

Por otro lado, el modo *Inmediate* es de menor nivel de programación y por lo tanto, todo el trabajo referente al manejo de la geometría interna 3D debe realizarlo el propio programador, lo que obliga a éste a tener que disponer de los conocimientos técnicos de lo que quiera realizar.

En la tabla 3 tenemos ejemplos de funciones disponibles en el API 3D, donde se puede observar que la forma de definirlas se parece mucho en todas las áreas del Direct X.

Direct Input

El API *Direct Input* incluye una serie de funciones que nos permiten controlar la entrada de datos provinientes de todo tipo de dispositivos de control, como son el teclado, el mouse, el joystick y otros periféricos más modernos, como podrían ser, por ejemplo, unos guantes de realidad virtual.

Tabla 3. Algunas funciones que aparecen en el API 3D.

IDirect3DRM::CreateObject
IDirect3DRM::CreateFrame
IDirect3DRM::GetDevices
IDirect3DRMPickedArray::GetPick
IDirect3DRMFrame::GetChildren
IDirect3DRMFrame::GetLights
IDirect3DRMFrame::GetVisuals
IDirect3DRMViewport::Pick
IDirect3DRMAnimation::AddPositionKey
IDirect3DRMAnimation::AddScaleKey
IDirect3DRMAnimation::SetTime
IDirect3DRMAnimation::SetTime
IDirect3DRMAnimationSet::AddAnimation
IDirect3DRMAnimationSet::DeleteAnimation
IDirect3DRMAnimationSet::SetTime
Direct3DRMMeshBuilder::GetFaces
IDirect3DRMDevice::InitFromClipper
IDirect3DRMDevice::QueryInterface

Este apartado de programación, aunque parezca menos importante, es también fundamental tenerlo a nuestra disposición puesto que todos conocemos la lentitud de las rutinas de acceso de joystick y teclado, por ejemplo, que en una aplicación como arcade 3D o 2D, donde las llamadas a las funciones de posicionamiento se ejecutan miles de veces, pueden ralentizar todo el sistema de forma considerable.

El uso de este API es muy simple, como muestra, realizando una simple llamada a la función *joyGetNumDevs* se nos devuelve el número de Joysticks instalados en el sistema, llamando a *JoyGetPosEx* podemos saber si un joystick está o no conectado y con joyGetDevCaps podemos obtener las funcionalidades de que disponemos en el mismo.

Con todo ello, el manejo de los diferentes tipos de teclado, ratones y otros punteros que se nos pueden presentar en el sistema se simplifica enormemente.

En la tabla 4 observamos la variedad de ratones y teclados que se soportan en este API.

Direct Sound

En el apartado de sonido, el API *Direct Sound* nos permite dentro de las operaciones típicas disponer de mezclador de sonido de alta velocidad, acelerar las operaciones de sonido por hardware y acceder, en resumen, al dispositivo de salida de forma directa. También, como pasa con el resto de APIs, todo aquello que no se pueda realizar por hardware se podrá realizar por emulación software.

La capacidad que quizás es más innovadora dentro del apartado de audio, es la posibilidad de gestionar sonido tridimensional, lo que realmente nos resulta imprescindible si estamos desarrollando un programa 3D de calidad que queramos que sea realmente interactivo como podría ser un entorno de realidad virtual.

Como era de esperar, las opciones dentro de este apartado son bastante completas, pudiendo controlar las coordenadas en el espacio de las fuentes de sonido, el movimiento de la fuente respecto al oyente y los conos de proyección del mismo (para controlar los volúmenes del sonido según la posición dentro del mismo en el espacio).

Direct Sound permite tanto gestionar sonido 2D como 3D

Como ejemplo sencillo se ha incluido una rutina en C++ que hace uso de los recursos del API *Direct Sound* y donde se crea un búfer de sonido con unas características determinadas.

Tabla 4. Identificadores que puede devolver el API según el tipo de ratón y teclado instalados en el sistema.

Tipos de ratón instalado:
DIDEVTYPEMOUSE_UNKNOWN: Tipo indeterminado
DIDEVTYPEMOUSE_TRADITIONAL: Ratón tradicional
DIDEVTYPEMOUSE_FINGERSTICK: Fingerstick.
DIDEVTYPEMOUSE_TOUCHPAD: Touchpad (portatil).
DIDEVTYPEMOUSE_TRACKBALL: Raton tipo TrackBall

Tipos de teclado instalado:
DIDEVTYPEKEYBOARD_PCXT: Teclado IBM PC/XT 83-teclas
DIDEVTYPEKEYBOARD_OLIVETTI: Teclado Olivetti 102-teclas.
DIDEVTYPEKEYBOARD_PCAT: Teclado IBM PC/AT 84-teclas.
DIDEVTYPEKEYBOARD_PCENH: Teclado Microsoft Natural.
DIDEVTYPEKEYBOARD_NOKIA1050: Teclado Nokia 1050.
DIDEVTYPEKEYBOARD_NOKIA9140: Teclado Nokia 9140.
DIDEVTYPEKEYBOARD_NEC98: Teclado Japones NEC PC98.
DIDEVTYPEKEYBOARD_NEC98LAPTOP: Teclado Japones NEC PC98 laptop.
DIDEVTYPEKEYBOARD_NEC98106: Teclado Japones NEC PC98 106-teclas.
DIDEVTYPEKEYBOARD_JAPAN106: Teclado Japones 106-teclas.
DIDEVTYPEKEYBOARD_JAPANAX: Teclado Japones AX teclas.
DIDEVTYPEKEYBOARD_J3100: Teclado Japones J3100 teclas.

Direct Play

Este API, aunque parezca destinado a la generación de sonido o algo parecido, se trata realmente de un conjunto de rutinas pensadas para facilitar la conectividad de nuestras aplicaciones, lo cual se usa mucho actualmente en la creación de videojuegos tipo Multiplayer (como es el famoso QUAKE, que tanto furor causa en las universidades).

El API de sonido 3D no permite fuentes estereofónicas de sonido

Las ventajas derivadas de usar este API destacan sobretodo porque el programador no tiene que preocuparse por el funcionamiento interno de los sistemas

Ejemplo de un programa que crea un búfer secundario de sonido para usar con el API

```
BOOL Crear_Buffer(
LPDIRECTSOUND lpDirectSound,
LPDIRECTSOUNDBUFFER *lplpDsb)
{
 PCMWAVEFORMAT pcmwf;
 DSBUFFERDESC dsbdesc;
 HRESULT hr;
 // Crea la estructura con el formato de sonido.
 memset(&pcmwf, 0, sizeof(PCMWAVEFORMAT));
 pcmwf.wf.wFormatTag = WAVE_FORMAT_PCM;
 pcmwf.wf.nChannels = 2;
 pcmwf.wf.nSamplesPerSec = 22050;
 pcmwf.wf.nBlockAlign = 4;
 pcmwf.wf.nAvgBytesPerSec =
  pcmwf.wf.nSamplesPerSec * pcmwf.wf.nBlockAlign;
 pcmwf.wBitsPerSample = 16;
 // Pone en uso la estructura DSBUFFERDESC:
 memset(&dsbdesc, 0, sizeof(DSBUFFERDESC)); // Zero it out.
 dsbdesc.dwSize = sizeof(DSBUFFERDESC);
 // Valores de control pan, volumen y frequencia:
 dsbdesc.dwFlags = DSBCAPS_CTRLDEFAULT;
 // 3-segundo búffer.
 dsbdesc.dwBufferBytes = 3 * pcmwf.wf.nAvgBytesPerSec;
 dsbdesc.lpwfxFormat = (LPWAVEFORMATEX)&pcmwf;
 // Crea el búffer.
 hr = lpDirectSound->lpVtbl->CreateSoundBuffer(lpDirectSound,
  &dsbdesc, lplpDsb, NULL);
 if(DS_OK == hr) {
  // Exito. Puntero a interface valido en *lplpDsb.
  return TRUE;
 } else {
  // Error.
  *lplpDsb = NULL;
  return FALSE;
 }
}
```

con los que se comunicará el programa con el exterior, sino que sólo necesitará aprender a usar la forma en que funciona la interfaz que posee el propio API, siendo todo el resto del proceso transparente al programador, tanto si la comunicación es por Internet, por cable o de otra forma.

Como esquema de funcionalidades del API, podemos dar la siguiente lista de características:

1- Soporte para Internet.
2- Soporte para conexión por *Direct serial.*
3- Soporte de *Unicode* para un mejor soporte de localización.
4- Capacidad de migración de Host. En caso de que el jugador que hace de host quede fuera de línea, el Host para automáticamente a otro jugador que se encuentre en línea.
5- Capacidad de las aplicaciones de comunicar el *Lobby-program,* que consiste en que un espectador puede recibir el estado actual de una partida (por ejemplo) y de sus condiciones iniciales sin haber estado conectado desde el inicio de la sesión (partida, para entendernos).

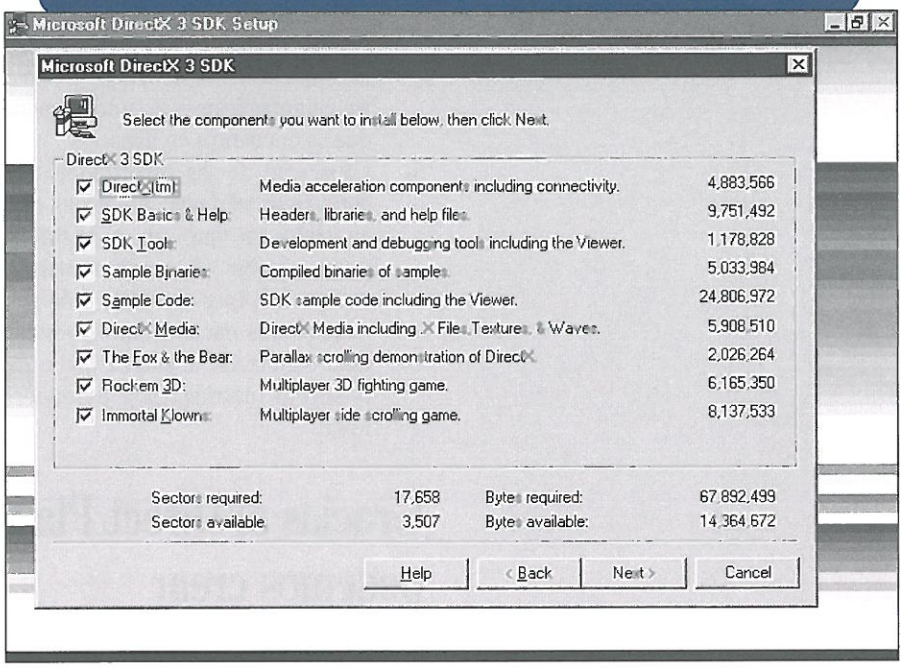
Gracias al Direct Play podemos crear videojuegos con soporte Multiplayer

6- Posibilidad de ejecutar más de una aplicación simultáneamente con uso de comunicación tipo Host.
7- Capacidad de detectar cuándo se produce una desconexión del host de una aplicación externa para poder actual al respecto.

El DirectX SDK

Una vez conocemos qué son las DirectX, podemos pasar a explicar lo referente al Kit de desarrollo, el DirectX SDK. Básicamente se trata de un kit creado por Microsoft para animar a los desarrolladores de aplicaciones (principalmente de videojuegos) a que utilicen el API DirectX en la creación de sus aplicaciones Windows. Con ello, Microsoft consigue dos objetivos al mismo tiempo, por un lado promover la creación de videojuegos bajo Windows y, por otro, convertir al DirectX en un API estándar en el mercado, con los beneficios que esto conlleva.

Pantalla de selección de elementos a instalar del programa instalador del DSK 3.0 .


Microsoft DirectX 3 SDK Setup
Microsoft DirectX 3 SDK
Select the components you want to install below, then click Next.
DirectX 3 SDK
DirectX(tm) Media acceleration components including connectivity. 4,883,566
SDK Basics & Help Headers, libraries, and help files. 9,751,492
SDK Tools Development and debugging tools including the Viewer. 1,178,828
Sample Binaries Compiled binaries of samples. 5,033,984
Sample Code: SDK sample code including the Viewer. 24,806,972
DirectX Media: DirectX Media including X Files, Textures, & Waves. 5,908,510
The Fox & the Bear: Parallax scrolling demonstration of DirectX. 2,026,264
Rockem 3D: Multiplayer 3D fighting game. 6,165,350
Immortal Klowns Multiplayer side scrolling game. 8,137,533
Sectors required: 17,658
Sectors available 3,507
Bytes required: 67,892,499
Bytes available: 14,364,672
Help
< Back
Next >
Cancel

Instalación del KIT

Una vez ejecutamos el programa SETUP.EXE del disco (de CDROM), éste se encarga de ejecutar el famoso *InstallShield Wizzard* del propio Windows para hacer más estándar y cómodo todo el proceso. Una vez dentro de él, se nos pregunta en qué directorio queremos instalar el Kit para, a continuación, poder elegir entre instalar todo el Kit completo, crear una instalación a medida (sólo los componentes que le indiquemos) o instalar sólo las propias Librerías DirectX.

Con WEB-link, podemos establecer vínculos con sitios internet relacionados con el tema

Una vez seleccionado el tipo de instalación que queremos realizar, debemos seleccionar si queremos instalar una versión de Depurado o completa del API DirectX. Esta opción es muy importante, ya que la versión de depuración hace que dispongamos de ayudas para encontrar los fallos de ejecución a la hora de testear los programas que realicemos.

Tras contestar a todo lo dicho hasta ahora aparecerán, en caso de que hayamos elegido instalar la versión a medida, los nueve elementos que conforman el Kit para poder seleccionar los que deseemos instalar.

En total, estos elementos ocupan un total de 64 Mbytes y son los que siguen:

1- **DirectX:** Los propios API's.
2- **SDK files & help:** Que son los propios ficheros de programación y ficheros de ayuda.
3- **SDK Tools:** Diversos programas destinados a ayudarnos en la depuración y desarrollo.
4- **Sample Binaries:** Los ejemplos incluidos ya compilados de todos los programas de muestra.
5- **Sample Code:** Los fuentes C/C++ de todos los ejemplos de muestra.
6- **DirectX Media:** Que contiene ficheros de ejemplo como son texturas y sonido.
7- **The Fox & The Bear:** Programa de ejemplo completo del uso de un Scroll Parallax.
8- **Rockem 3D:** Ejemplo de un juego de lucha en 3D con personajes vectoriales 3D.
9- **Immortal Klowns:** Ejemplo de un juego simple en 2D.

Como vemos, de todos los componentes mostrados, tan sólo es necesario tener algunos instalados (1-3) mientras que el resto (sobre todo los 3-4 últimos), son únicamente ejemplos ejecutables.

Los ficheros necesarios para la programación

A parte del propio API DirectX, el componente número 2 es básico que sea instalado para poder empezar la programación X ya que, una vez en nuestro disco, dispondremos de los ficheros de cabecera .H tan necesarios para la definición de las funciones externas en nuestros propios fuentes, con lo cual ya tendremos acceso al DirectX desde nuestros propios compiladores de C y C++.

Por otro lado, todo el sistema de ayuda se instala como parte del elemento de instalación 2, con lo cual disponemos de los ficheros que hacen las veces del manual del programador y que son bastante completos y detallados. Como dato a resaltar, decir que no se instalan sólo ayudas en formato típico de Windows (ficheros .HLP), sino que también dispondremos de casi seis megas de documentos Word (un verdadero libro) y de un pequeño conjunto de ficheros .HTM de navegación WEB que incorporan enlaces con otros puntos de referencia relacionados sobre el tema, como documentos concretos que hay en la WEB Microsoft, vínculos a CDROMs, e incluso juegos.

Complementos del programador

Como complementos de testeo, depurado y ejemplos, disponemos de diversos pequeños programas que nos pueden ayudar en aspectos diferentes. A continuación vamos a detallar a los principales:

DirectX Viewer: éste es uno de los principales programas que acompañan a este KIT. Se trata de un visor donde se muestran todos los recursos del sistema del API DirectX organizado todo ello en forma de árbol donde cada API es un directorio y dentro de cada uno podemos acceder a todos sus apartados. Así tenemos, por ejemplo, que dentro del apartado DIRECT 3D devices, sólo dentro del subárea RGB, aparecen los siguientes apartados: Color Model, DevCaps, TransformCaps, Lighting Caps, Bcliping, LineCaps, Tricaps, Misc... dentro de cada uno de los cuales podemos ver todo lo referente a su estado de funcionamiento.

DirectX DSK Help: como hemos dicho, el kit trae toda una colección de ficheros de ayuda tipo .HLP que podemos visualizar desde este programa y donde se encuentran los contenidos estructurados en el típico estilo de los sistemas de ayuda Windows95, con temarios que se pueden abrir para poder acceder a sus contenidos por separado.

DirectDraw Test: con este programa podemos testear el funcionamiento del API Direct Draw y realizar las principales operaciones desde su menú de opciones.

DirectDraw Caps: este programa funciona como complemento del anterior y su función es la de poder probar todos los modos gráficos disponibles así como poder alterar en tiempo real en cualquier momento las banderas de control del API Direct Draw, las cuales gobiernan el funcionamiento de Paletas, efectos especiales, Color Keys y de todos los parámetros que es posible modificar.

Los ejemplos incluidos en el Kit permiten ver el rendimiento de Direct X

Direct3D test: con este programa dispondremos de un Testeador de visualización y de tiempos de ejecución de tres ejemplos de animaciones 3D y de muchos bucles de ejecución incorporados de ejemplo. Los testeos de visualización 3D pueden adaptarse con las características generales que deseemos, pudiendo elegir entre otras opciones: rellenado Flat, tipo Gouraud, si queremos texturas, tipo de filtros a aplicar, etc. Los testeos de tiempos ejecución de bucles son un total de diez, e incluyen velocidad de transformación de vértices, tiempos de renderizado de triángulos, creación de texturas RGB, etc. Al final de cada test se nos ofrece el resumen con el número de iteraciones realizadas de media por segundo.

Direct3D Viewer: este programa es básicamente un visualizador de objetos tridimensionales almacenados en ficheros tipo .X Como pasaba en el anterior, para realizar las pruebas de visualización podemos modificar todos los parámetros de renderizado (Mono/RGB, Tipo rellenado polígonos, etc.) e incluso las características de las luces. Este programa nos puede ayudar, por ejemplo, a realizar los cálculos de velocidad a la hora de diseñar escenas 3D en movimiento de tiempo real con unas características de calidad de renderizado X en para nuestras aplicaciones.

DirectSound Mixing Test: este programa es un sencillo mezclador con el que tenemos la posibilidad de cargar varios sonidos en memoria y reproducirlos todos simultáneamente.

DirectSound Streaming Test: permite la reproducción de un sonido digitalizado al que se le puede cambiar la frecuencia de muestreo, el volumen y el efecto *Pan* en cualquier momento durante su reproducción.

La renderización en 3D se puede realizar también emulando color RGB

FullDuplex Audio Sample: permite la reproducción y la grabación de sonido simultáneamente en y desde la tarjeta de sonido, lo cual se conoce actualmente como FullDuplex (término usado también en modems que permiten entrada y salida de datos a la vez).

MidiStreaming Test: como su nombre indica, es una utilidad para la reproducción de ficheros musicales tipo .MID.

Otros: el resto son básicamente demos de escenas 3D y 2D que permiten que el programador pueda verel funcionamiento de las rutinas DirectX en la práctica.

Para terminar, sólo recordar que todos los fuentes que contiene el Kit, incluyendo los programas mencionados (Visor X, Direct 3D Viewer, etc.), están en forma de fuentes C y C++, y en ellos el programador puede comprobar la manera en que funciona el código, lo cual puede ser de mucha utilidad para aquellos que se inician en la programación de estos APIs.

Visión estereoscópica: la cuarta dimensión

Manel Guerrero

La creación de mundos virtuales por ordenador hace mucho que dejó de ser una utopía. En la actualidad disponemos de programas que nos sumergen en espacios tridimensionales en los que la sensación de realismo es casi total. Las técnicas utilizadas para generar dichos entornos revolucionaron hace algunos años el mundo de la informática personal.

Ahora una nueva revolución en la generación de entornos virtuales está a punto de llegar a nuestras casas. Y esa revolución lleva el nombre de 'visión estereoscópica'. La visión estereoscópica nos permitirá alcanzar nuevas cotas de realismo hasta ahora insospechadas. Con ella "veremos" como los objetos tridimensionales generados por nuestro ordenador adquieren volumen y "salen" del monitor, ofreciéndonos la sensación de que están flotando frente a nosotros.

En este artículo aprenderemos a crear programas capaces de proyectar objetos tridimensionales mediante visión estereoscópica.

La perspectiva cónica frontal

Antes que nada, debemos repasar cómo funciona la perspectiva cónica frontal. Como podemos ver en la figura 1, dicha perspectiva se caracteriza por un plano de proyección (P) sobre el cual intersectan las líneas de visión que parten del punto de visión (PV) hasta los objetos a representar.

Esta perspectiva es la misma que obtenemos si miramos a cualquier objeto con un sólo ojo.

La visión estereoscópica

El siguiente paso son las estereoscopías. La visión estereoscópica consiste en ofrecer dos imágenes distintas, una para cada ojo, con el fin de generar una sensación de volumen. Esto se consigue mediante dos perspectivas cónicas frontales del mundo tridimensional a representar.

En la figura 2 podemos observar cómo dispondremos dichas perspectivas con el fin de emular nuestro sistema de visión.

Las dos perspectivas tienen los puntos de visión (PVD y PVI) separados por la distancia interocular (dio) y sus ejes principales de proyección intersectan en un punto de enfoque (PE). En la figura podemos observar la relación entre la distancia focal (df), la distancia interocular y el ángulo de enfoque (a).

Dispositivos de visión estereoscópica

Ahora que ya hemos visto en qué consiste la visión estereoscópica, vamos a ver los diferentes dispositivos que nos permiten disfrutar de ella:

• El casco virtual (o casco 3D). Este dispositivo consiste en una especie de casco en el que se han instalado dos pantallas, que suelen ser de cristal líquido, de tal forma que cada ojo ve una de las pantallas. Ofrecen buenos resultados. Sus peores inconvenientes son su precio y la poca resolución de sus pantallas.

• Las gafas sincronizadas por frecuencia. Este dispositivo consiste en unas gafas que están sincronizadas con el monitor de tal forma que cuando el monitor ofrece la imagen correspondiente al ojo derecho la lente izquierda de las gafas no deja pasar la luz y viceversa. Ofrece resultados bastante buenos, siempre que consigamos una frecuencia de sincronización aceptable.

• Las gafas por filtro de color. En este caso las dos imágenes se muestran simultáneamente en el mismo monitor y son los filtros de color que incorporan las gafas los que hacen que a cada ojo le llegue una imagen distinta. Este artículo se centrará en este último dispositivo, ya que es el más accesible para los lectores. Su inconveniente es que los objetos representados se verán de forma monocromática.

Figura 1

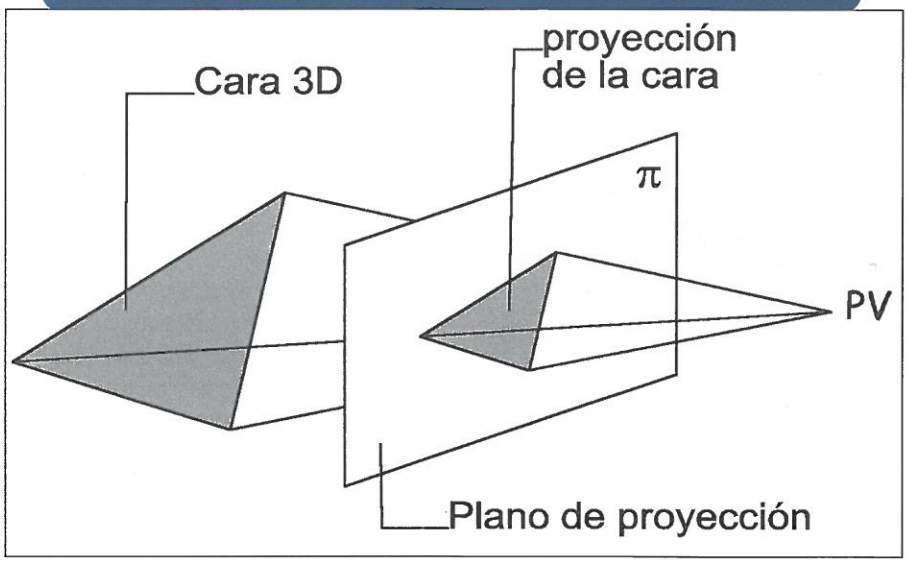
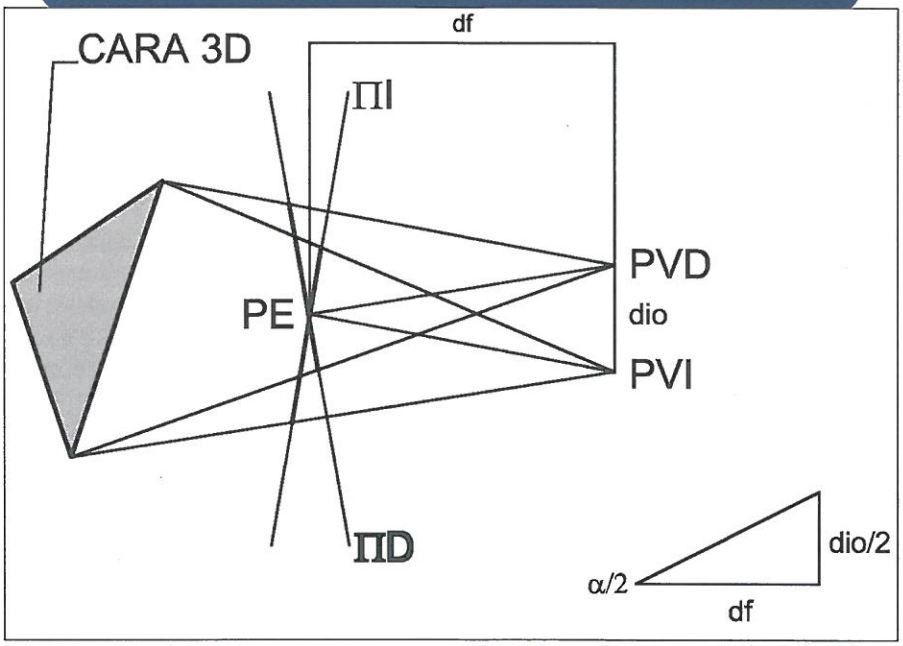


Figura 2



Gafas por filtro de color

La codificación de color RGB para las dos lentes de las gafas es la siguiente:

• [0%,0%,100%] lente derecha.
• [100%,0%,0%] lente izquierda.

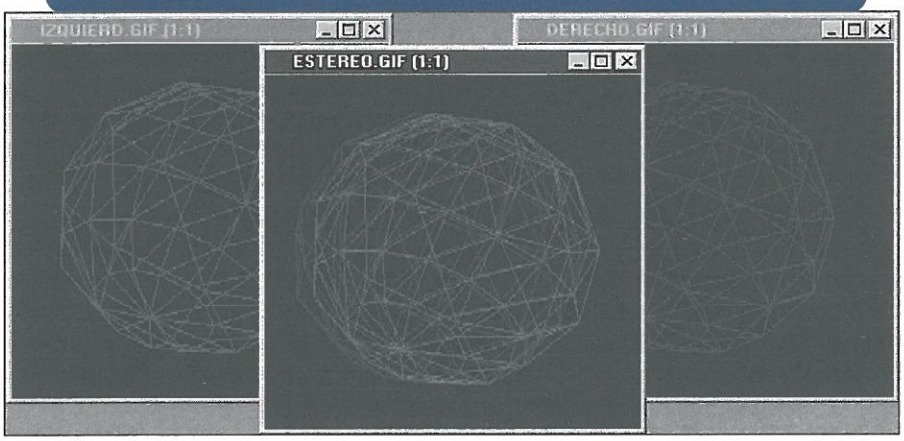
Lo que queramos que vea el ojo derecho deberemos dibujarlo en azul (0%,0%,100%), lo que queramos que vea el ojo izquierdo en rojo (100%,0%,0%) y lo que queramos que vean los dos ojos en morado (100%,0%,100%). En la figura 3 podemos observar cómo funcionan las gafas por filtro de color.

Cálculo de la proyección cónica frontal

Llegados a este punto, ya disponemos de suficiente información para ver cómo se calculan las proyecciones. En la figura 4 podemos ver las fórmulas utilizadas para calcular la proyección de los puntos situados en un espacio tridimensional para poderlos representar en un monitor.

En dichas fórmulas el punto tridimensional (x,y,z) es proyectado sobre el plano frontal en la coordenada (sx,sy). De manera que ya hemos conseguido transformar las coordenadas de un objeto de tres dimensiones a las dos dimensiones de nuestro monitor.

Figura 3



El signo menos que aparece en la ecuación que calcula el valor de 'sy' es debido a que en el monitor el eje vertical tiene sentido opuesto al del espacio cartesiano.

Para dibujar la proyección de cualquier objeto tridimensional primero calcularemos las proyecciones de sus puntos y luego trazaremos las aristas, o rellenaremos las caras, del objeto en cuestión utilizando las coordenadas de proyección.

Figura 4. Proyección cónica frontal.

$$sx = \frac{x}{1 - (z/df)} \qquad sy = -\frac{y}{1 - (z/df)}$$

Cálculo de la proyección estereoscópica

Una vez comprendidas las fórmulas de proyección cónica y el método de construcción de la visión estereoscópica es fácil calcular cuales serán las fórmulas de proyección estereoscópica. Éstas aparecen en la figura 5. Lo que estamos haciendo en ellas es desplazar el punto de vista (PVD y PVI) sobre el eje 'Y' la mitad de la distancia interocular (dio) y rotar el eje de visión sobre el plano de deriva ('XZ') la mitad del ángulo de enfoque (a). Así logramos emular la posición de nuestros ojos cuando miran a un objeto (ver figura 2). El punto (sxd,syd) corresponde a la proyección del objeto que ve el ojo derecho y (sxi,syi) la que ve el izquierdo.

Para dibujar la proyección de un objeto tridimensional, de nuevo, primero calcularemos las proyecciones de sus puntos para los ojos derecho e izquierdo. Después trazaremos las aristas, o rellenaremos las caras, en azul de la proyección para el ojo derecho. Y finalmente haremos lo mismo, pero en rojo, con la proyección para el ojo izquierdo. Teniendo en cuenta que los píxeles a los que corresponda pintar tanto en rojo como en azul, deberán pintarse de color morado (ver tabla 1).

Cuando los objetos se mueven

Sin embargo, representar objetos en un espacio tridimensional de forma estereoscópica no es suficiente. Debemos permitir que se muevan en dicho espacio. Esto lo conseguiremos aplicando transiciones precalculadas de las matrices de transición para tres dimensiones. En la figura 6 podemos ver dichas matrices y en el código fuente observamos cómo precalculamos el producto de cada una de esas con los puntos del objeto.

Figura 5. Proyección estereoscópica.

$$td = 1 - \frac{(z+df) * \cos(\alpha/2) - x * \sin(\alpha/2) - df}{df}$$

$$sxd = \frac{(z+df) * \sin(\alpha/2) - x * \cos(\alpha/2)}{td}$$

$$syd = -\frac{y}{td}$$

$$ti = 1 - \frac{(z+df) * \cos(-\alpha/2) - x * \sin(-\alpha/2) - df}{df}$$

$$sxi = \frac{(z+df) * \sin(-\alpha/2) - x * \cos(-\alpha/2)}{ti}$$

$$syi = -\frac{y}{ti}$$

No nos extenderemos más en el tema de las transformaciones tridimensionales puesto que este tema ya ha sido tratado en otras ocasiones en esta revista.

El programa de ejemplo

Una vez ya hemos visto lo simple que es generar estereoscopías, vamos a ver un programa de ejemplo que nos permite ver una animación de objetos en visión estereoscópica a la vez que nos permite aplicarle las traslaciones, rotaciones y escalaciones ya vistas.

El programa está dividido en seis módulos unidos por un fichero proyecto. En el código fuente está explicado el funcionamiento de cada una de las diferentes subrutinas. A continuación explicaremos la función de cada uno de los módulos:

Tabla 1. Colores adecuados a los filtros de las gafas estereoscópicas.

Ojos	Color	Color RGB	Paleta 16 colores
Derecho	azul	0%,0%,100%	1
Izquierdo	rojo	100%,0%,0%	4
Ambos	morado	100%,0%,100%	5

• ANI-3D.C: módulo principal. En él se definen los diferentes menús que aparecen en el programa, se gestionan los errores, se generan la sucesión de proyecciones y se calculan las traslaciones, rotaciones y escalaciones que el usuario ordene.

• _ASC.C: módulo que permite leer determinados tipos de objetos tridimensionales en formato 'asc'. Lo utilizaremos para leer los objetos que vamos a proyectar.

• _MENUS.C: módulo que implementa el sistema de ventanas, menús y botones que se utilizan en el programa.

• _MOUSE.C: éste es el típico módulo del mouse. En él se implementan funciones como: mostrar y ocultar mouse, saber en que posición está, moverlo a una nueva posición, etc. Si el lector aún no sabía cómo hacer que su programa usara el mouse, puede sentirse libre para utilizarlo a partir de ahora.

• _PROJEC.C: en este módulo se implementan las diferentes proyecciones con que cuenta el programa: vista plana, perspectiva cónica frontal (figuras 1 y 4) y visión estereoscópica (figuras 2 y 5).

• GRAPH3D.C: en este módulo sólo hay una función. Dicha función permite (utilizando el algoritmo de Bresenham) dibujar líneas rectas de tal forma que, en función de que color tenía cada uno de los píxeles de la recta, éstos se pinten de un color o de otro. Se utiliza para (una vez hemos dibujado la vista en azul) dibujar cada píxel de la otra vista en rojo si aún no estaba pintado y en morado si ya estaba pintado en azul. Esta función debería implementarse en ensamblador para obtener mayor velocidad.

Ejecutando el programa

Al ejecutar el programa aparece una ventana de presentación. Pulsando el botón 'Ok' iremos al menú principal. En él pulsaremos la opción 'Animación 3D' e iremos a un nuevo menú donde se nos permitirá elegir el nombre de la animación y el número de fotogramas de que está compuesta (figura 7). Para ello pulsaremos en el campo que queremos editar, modificaremos el valor del campo y pulsaremos 'enter' para confirmar. Hay preparadas 3 animaciones 'A1', 'A3' y 'A5' (todas ellas de veinte fotogramas).

Una vez hecho esto, pulsaremos el botón de 'Ok' y se nos informará de las teclas de que disponemos para controlar la

Figura 6. Matrices de transición en un espacio tridimensional.

$$\text{Transición: } \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

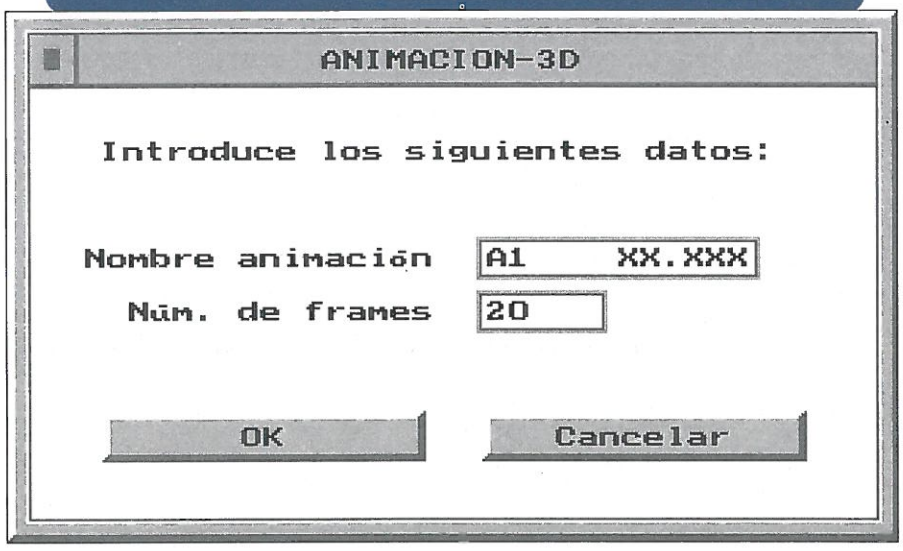
$$\text{Rotación X: } \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 0 & \sin 0 & 0 \\ 0 & -\sin 0 & \cos 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotación Y: } \begin{bmatrix} \cos 0 & 0 & -\sin 0 & 0 \\ 0 & 1 & 0 & 0 \\ \sin 0 & 0 & \cos 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotación Z: } \begin{bmatrix} \cos 0 & \sin 0 & 0 & 0 \\ -\sin 0 & \cos 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Escalón: } \begin{bmatrix} ex & 0 & 0 & 0 \\ 0 & ey & 0 & 0 \\ 0 & 0 & ez & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 7



animación (figura 8). Ya solo falta pulsar cualquier tecla y ver la animación.

Si el ordenador donde lo ejecutamos no es lo suficientemente potente puede que los fotogramas se refresquen con lentitud. Esto podría solventarse optimizando la función 'xline' del modulo '_graph3d.c'.

Mientras se van sucediendo los diferentes fotogramas podremos trasladar, rotar y escalar el objeto, parar la animación y cambiar el tipo de perspectiva. Para salir de la animación basta con pulsar 'Esc'. Al principio puede que nos cueste un poco familiarizarnos con las teclas de control. Podemos probar lo siguiente: entrar en el menú 'Animación 3D', pulsar el botón 'Ok' y luego cualquier tecla. Eso cargará la animación por defecto (A1). Ahora podemos aumentar el zoom tres veces (tecla '+') para que ocupe toda la pantalla. Girar el objeto sobre el eje Z en sentido negativo unas tres veces (tecla 'H'); y otras tres veces sobre el eje X en sentido positivo (tecla 'R'). Podemos detener la animación pulsando 'espacio' y comprobar que el programa permite aplicar transformaciones aunque la animación esté detenida. Pulsaremos de nuevo 'espacio' cuando queramos que siga la animación. Si nos liamos y queremos volver a la disposición inicial, pulsaremos la tecla cero.

Conclusiones

En este artículo hemos aprendido a utilizar en nuestros programas la visión estereoscópica. Desde aquí quiero animar a los lectores a que 'jueguen' con el código del programa, lo cambien y experimenten con él. Y, quién sabe, puede que alguno nos sorprenda en breve con un nuevo juego del estilo de 'Doom', pero esta vez incorporando las estereoscopías.

Artículos de Sólo Programadores relacionados con el tema

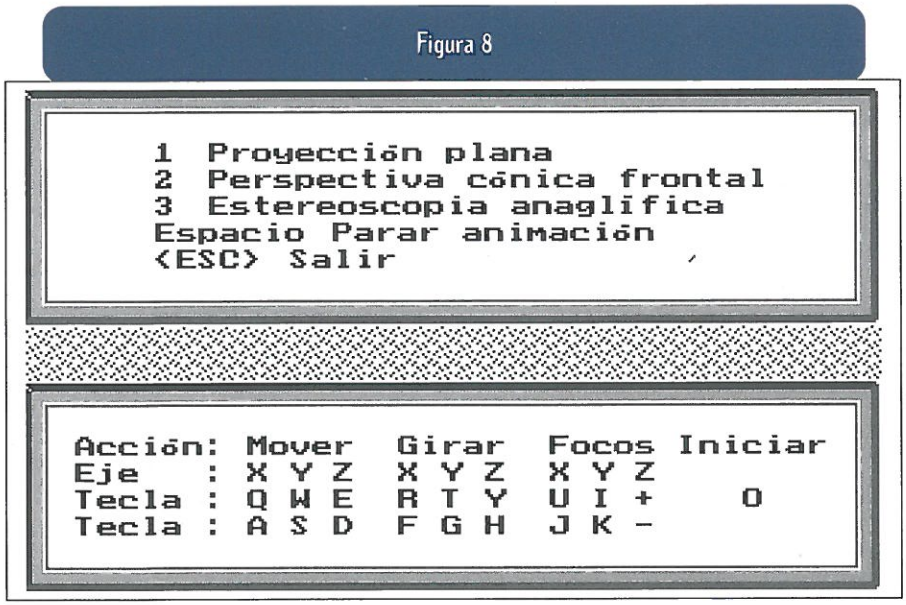
TRASLACIONES, ROTACIONES Y PROYECCIONES EN 3D

- Luis Fernando Fernández: "Creación en 3D" (Sólo Programadores 1).
- Luis Fernando Fernández: "Giros en el espacio" (Sólo P 2).
- Pedro Antón: "Introducción a las 3D" (Sólo P 20).

PROGRAMACIÓN DEL MOUSE

- José Ramiro: "Control del ratón" (Sólo P 26).

Figura 8


1 Proyección plana
2 Perspectiva cónica frontal
3 Estereoscopia anaglífica
Espacio Parar animación
<ESC> Salir
Acción: Mover Girar Focos Iniciar
Eje : X Y Z X Y Z X Y Z
Tecla : Q W E R T Y U I + O
Tecla : A S D F G H J K -

Bibliografía

- Microsoft: "Guía para programación de ratones" Microsoft-Anaya Multimedia, 1991. (Todo sobre la programación del mouse).
- Lee Adams: "Gráficos en C y ensamblador" Anaya Multimedia, 1992. (Interesante, aunque algo obsoleta, introducción a la programación de gráficos).
- Donald Hearn y M. Pauline Baker: "Gráficos por computadora" Prentice Hall, 1995. (Excelente y extenso libro que abarca casi todo sobre gráficos por ordenador).
- Miguel Angel Rodríguez-Roselló: "8088-8086/8087 Programación ensamblador", 1993. (En él podemos encontrar el algoritmo de Bresenham)

Agradecimientos

La aplicación presentada en este artículo es una adaptación de una de las seis aplicaciones que componen mi proyecto final de carrera que defendí el pasado mes de abril. Es por ello que quiero de nuevo agradecer a todos los que contribuyeron a la buena consecución de este. Merecen mención especial los miembros de la ANT (*Asociació Noves Tecnologies* 'http://ant.upc.es'), Enrique Algara de Anaya Multimedia y mi ponente Manel López.

Para contactar con el autor

A través de correo electrónico: tarom@hotmail.com y tarom@ant.upc.es, y visitando sus páginas web en la red: http://www.geocities.com/SiliconValley/Heights/1285 y http://ant.upc.es/~tarom.

Primer contacto con Watcom C/C++, versión 11

Emilio Postigo Riancho

HERRAMIENTAS DE DESARROLLO

Con la nueva versión de este compilador, la firma Sybase suministra un producto con las herramientas y librerías necesarias para crear programas de 16 y 32 bits destinados a ejecutarse en los sistemas operativos más populares de la plataforma PC.

Multientorno

Al contemplar el pequeño paquete que contiene esta herramienta, entre la propaganda que estamos acostumbrados a encontrar, una palabra llama nuestra atención: *multiplatform*.

Si se es estricto, su traducción literal podría llevar al lector a la conclusión de que, en efecto, esta herramienta permite desarrollar en máquinas con procesadores pertenecientes a distintas familias. No es así. Aunque Watcom generará código optimizado para diversos micros (desde el 8086 al Pentium Pro), todos ellos pertenecen a la familia Intel.

La calificación más adecuada para este compilador sería, por lo tanto, la de multientorno, ya que permite codificar y depurar aplicaciones para diversos sistemas operativos sin necesidad de tenerlos instalados en el equipo desde el que se desarrolla. En este aspecto, Watcom 11 dispone del más amplio rango de entornos para la plataforma Intel x86. La tabla 1 muestra todos los entornos de desarrollo posibles y en cuáles se dispone de un entorno integrado de desarrollo (I.D.E.). Las tablas 2 y 3 presentan, por su parte, para qué entornos es posible desarrollar.

Compatibilidad con Microsoft

En general se ha procurado lograr la mayor compatibilidad posible con Microsoft, tanto en los aspectos de generación de código como en las características de las herramientas de desarrollo.

MFC

Se soportan las versiones 4.1 (32 bits) y 2.52b (16 bits) de *Microsoft Foundation Classes*, incluidas en el producto, y se dispone de dos conjuntos de estas librerías: las *de producción*, empleadas cuando se compila la versión definitiva de la aplicación, y las *de depuración*, que son las que deben ser empleadas para hacer posible el rastreo de errores dentro del código.

Junto con estas librerías de clases se suministra una gran cantidad de ejemplos que ilustran sus posibilidades. Estos

Mucho ha sido el esfuerzo invertido por Sybase en su último compilador de C y C++: programación multientorno, desarrollo rápido bajo Windows, mayor compatibilidad con Microsoft, máxima optimización del código generado... Estas son algunas de las características más destacadas de la nueva herramienta.

Tabla 1.

Entorno de desarrollo	I.D.E.	Línea de comandos
D.O.S.	No	Sí
OS/2 32-bits	Sí	Sí
Windows 3.x	Sí	No
Windows 95	Sí	Sí
Windows NT	Sí	Sí

ejemplos deben ser procesados mediante el archivo de proceso por lotes *MKSAMPLE.BAT* antes de poder ser utilizados.

Si nuestra aplicación maneja clases MFC y deseamos que sean visualizadas con el *Browser*, el proceso de compilación deberá realizarse empleando la opción "db".

Novedades del lenguaje

El compilador de C y el de C++ ha experimentado modificaciones con respecto a la versión anterior, Watcom 10.6, que redundan tanto en una mayor velocidad de ejecución del código resultante como en un desarrollo similar al que permiten habitualmente las herramientas de Microsoft.

Por lo que respecta al compilador de C++, éste reconoce un nuevo tipo de variable y varias palabras reservadas. El nuevo tipo es *bool* y, como es de esperar, permite declarar variables capaces de contener valores *true* y *false*.

Las nuevas palabras reservadas que encontramos son:

- *mutable*: permite declarar datos miembros en una clase, de manera que puedan ser modificados por funciones miembro de esa clase aunque éstas hayan sido definidas como de tipo *const*.
- *explicit*: indica que un constructor de una clase no será invocado en asignaciones que impliquen conversiones implícitas de tipos de datos.

La identificación de tipos en tiempo de ejecución (Run-Time Type Identification o RTTI) también está contemplada

Otra novedad que podemos observar con respecto a la versión anterior es el soporte de los *espacios de nombres* (*namespaces*). Mediante el empleo de este elemento del lenguaje, se hace posible fragmentar y organizar una aplicación en diferentes subsistemas que tienen ámbitos de visibilidad completamente independientes.

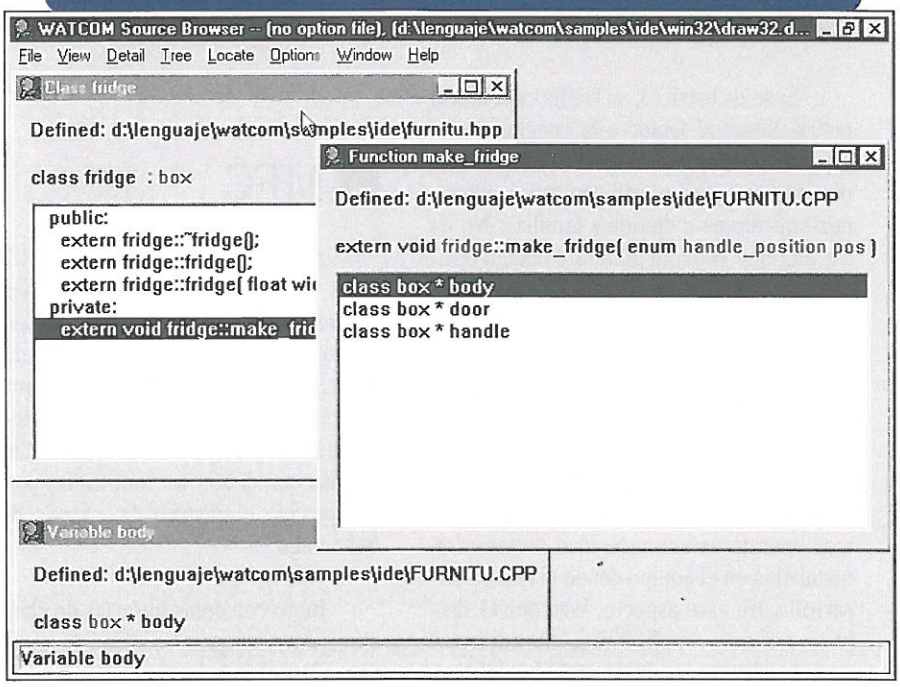
Tabla 2. Entornos destino de 16 bits.

D.O.S.
Windows 3.x
OS/2 1.x

Tabla 3. Entornos destino de 32 bits.

D.O.S. en modo protegido
Windows 3.x + extensor Watcom de 32 bits
Win32s
Windows 95
Windows NT
OS/2
Novell NLMs
AutoCAD ADS

Figura 1.



Pero hay más: la *identificación de tipos en tiempo de ejecución* (*Run-Time Type Identification* o *RTTI*) también está contemplada. Esta característica del lenguaje permite que un programa en ejecución pueda conocer el tipo de un objeto o variable, incluso cuando éstos son accedidos a través de un puntero o una referencia. Por último, se han incluido también los nuevos *cast* (conversores de tipo) de C++: *reinterpreter_cast, const_cast, static_cast* y *dynamic_cast*.

Tanto el compilador de C++ como el de C soportan archivos de cabecera precompilados, lo que redunda en un aumento de la velocidad de los procesos de compilación.

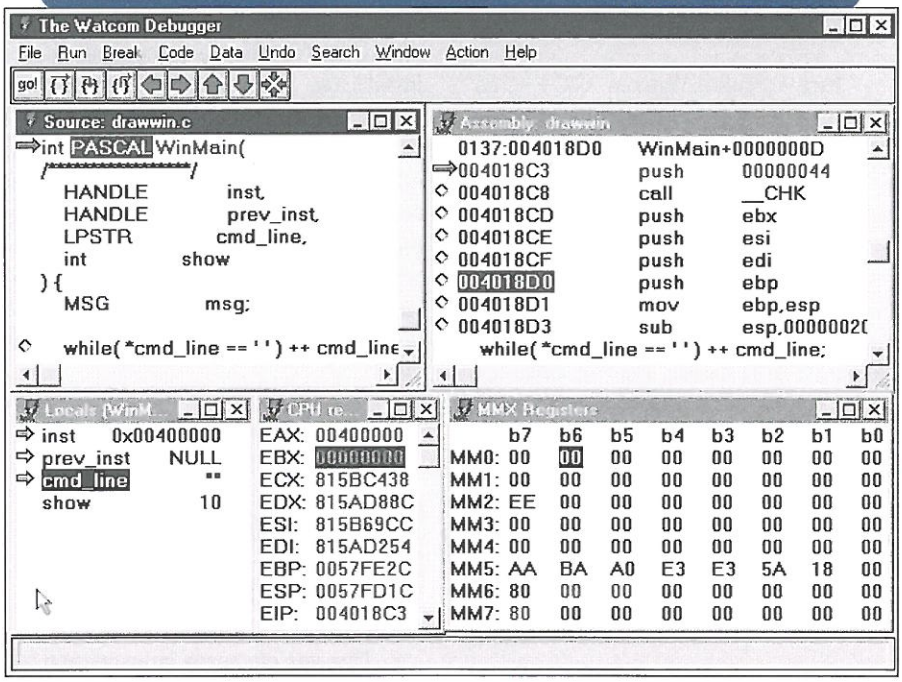
El compilador de C++ y el de C soportan archivos de cabecera precompilados

Por otra parte los programas enlazador (*Watcom Linker*) y gestor de librerías (*Watcom Library Manager*) pueden manejar tanto el formato AR, compatible con Microsoft, como los ficheros objeto de tipo COFF, ELF y OMF. Las librerías de C y C++ se han visto modificadas, de manera que ahora se puede contar con la posibilidad de manejar en el mismo *stream* variables multi-byte y wide.

Óptima generación de código

El código generado se encuentra preparado para aprovechar la lógica de predicción de saltos, especialmente en el caso del procesador Pentium Pro.

Figura 2.



Esta nueva ventaja se materializa empleando las opciones de compilación "ob" y "ox".

En este aspecto, la herramienta deja notable libertad al programador a la hora de producir el código más eficaz para cada procesador. Si bien las opciones propuestas por defecto representan, en opinión de Sybase, las más adecuadas, la imposibilidad de conocer de forma exacta cuál es la opción más apropiada en una circunstancia determinada invita a experimentar con las diferentes posibilidades. Aquí habrá que comportarse con cautela: el código más eficaz para un Pentium no lo será en todos los casos para un 386.

El programa WASM, el ensamblador de WATCOM, incluido en el paquete soporta las nuevas instrucciones MMX que INTEL ha añadido a sus procesadores

Ensamblador con soporte MMX

El programa WASM, el ensamblador de WATCOM, incluido en el paquete, soporta las nuevas instrucciones MMX que INTEL ha añadido a sus procesadores. Por otro lado se incluye una nueva versión de WDIS, programa desensamblador, que reemplaza al antiguo WDISASM.

También aquí se ha efectuado un esfuerzo por alcanzar compatibilidad con Microsoft: se han añadido herramientas (nmake, cl, link, lib,...) que emulan las de Microsoft y emplean sus mismas opciones de línea de comandos. Siempre que es posible, estas opciones se traducen a las correspondientes de Watcom y se invoca la herramienta propia.

Requerimientos del sistema

Para ejecutar Watcom C/C++ es necesario un PC con procesador 80386 o superior, al menos 8 MB de memoria RAM y un mínimo de 514 MB de espacio libre en disco.

Debido a que el producto se suministra en formato CDROM, es preciso disponer de la unidad lectora apropiada. Si se carece de la misma, pero se tiene acceso a alguna, el programa de instalación contiene la opción de crear un conjunto de discos de instalación de 3 _ pulgadas. El número de discos necesario dependerá de las opciones escogidas. Si se seleccionan todas (instalación completa) serán necesarios unos cien disquetes.

Instalación

El proceso de instalación arranca con la ejecución del programa SETUP.EXE, salvo si se trabaja desde OS/2, en cuyo caso debe ejecutarse INSTALL.EXE.

La instalación completa en disco de exige 1 Gb

La instalación completa en disco de este producto exige 1 Gb, cantidad que en este momento no puede considerarse desmesurada, pero que puede obligar a la mayoría de los desarrolladores a disponer de un disco duro sólo para este producto. Además de la instalación completa, se ofrece la posibilidad de dejar los archivos de ayuda en el CD-ROM, la de personalizar nuestra instalación y la de trabajar básicamente desde el CD-ROM. Aunque esta opción libera enormemente recursos de nuestro sistema, no es recomendable a no ser que se disponga de un lector CD de al menos óctuple velocidad.

Una vez que se haya finalizado la instalación, se crearán tres grupos de programas con las utilidades y los archivos de ayuda necesarios, y en este punto se presenta un pequeño inconveniente: será necesario incluir tales grupos "a mano" dentro del menú de inicio del sistema, debido a que el programa de instalación parece que no consulta el registro del sistema para localizar el menú de inicio.

Instalación incremental

Una vez efectuada la instalación, es posible agregar y/o quitar componentes de la misma. Para realizar estas operaciones es suficiente con marcar dentro del panel de opciones aquellos elementos que nos son necesarios y desmarcar los que no deseamos. Para desinstalar totalmente el producto se dispone de una opción específica.

Qué incluye esta herramienta

Sybase ha incorporado en Watcom 11 numerosas características que los programadores llevaban tiempo esperando. El entorno integrado de desarrollo (*I.D.E.*) permite realizar diversas operaciones como editar, compilar, enlazar y depurar aplicaciones para cualquier entorno de los que han sido anteriormente mencionados de forma sencilla y eficaz.

Los proyectos pueden tener como resultado distintas salidas (*targets*), lo cual permite generar, por ejemplo, EXEs y DLLs en un mismo proceso de compilación y enlace. El IDE genera archivos MAKE con formato texto, que pueden ser visualizados y editados con cualquier programa editor de texto.

Los proyectos pueden tener como resultado distintas salidas (targets)

Figura 3.

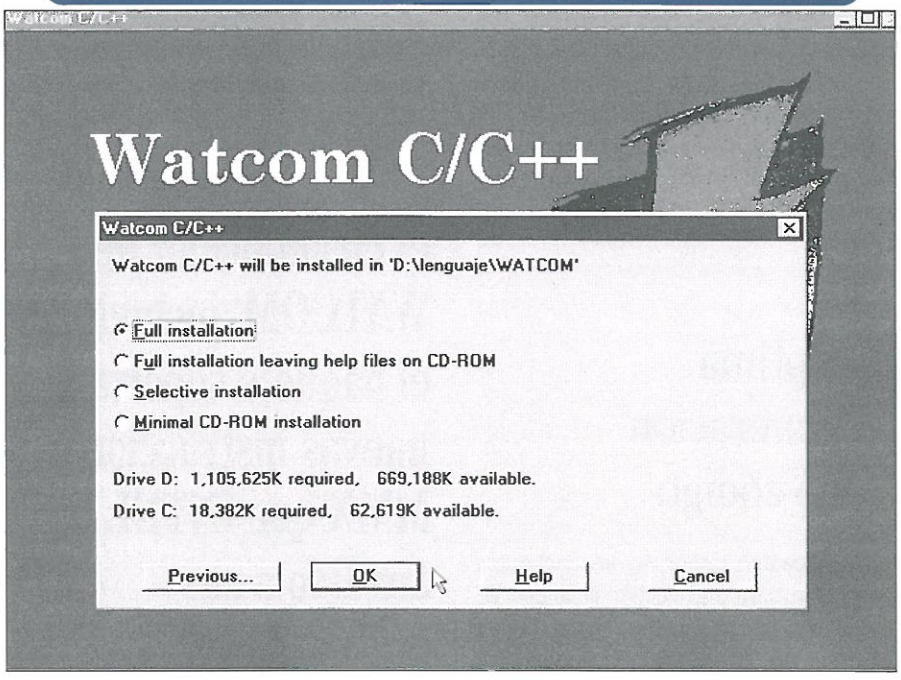
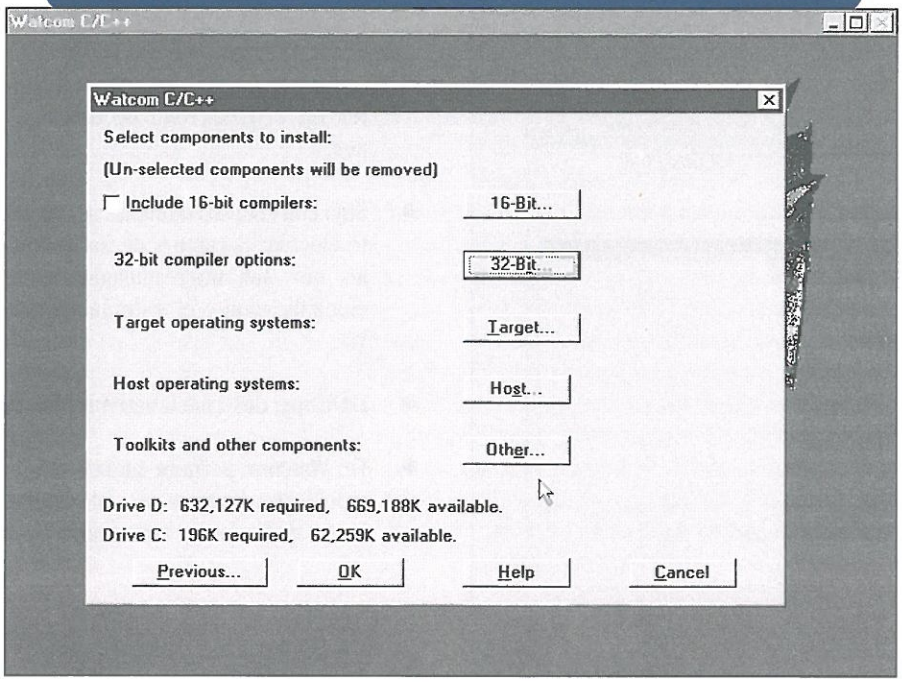


Figura 4.



Las herramientas que componen este producto están optimizadas para aprovechar las características más avanzadas de los sistemas operativos de 32 bits y los últimos procesadores aparecidos en el mercado.

Con este objetivo, se permite la generación de código para el Pentium MMX mediante la inclusión de instrucciones ensamblador *in-line* en el interior de un programa empleando la palabra reservada *_asm*.

Watcom está compuesto de un amplio conjunto de utilidades y herramientas gráficas que pretenden abarcar todas las necesidades de un desarrollador. Desde su entorno integrado será posible ejecutar desde un depurador, un visualizador de clases o diversas herramientas de análisis, hasta numerosas herramientas gráficas para crear recursos y, por supuesto, se puede utilizar un editor de código fuente que no tiene por qué ser el *Watcom Editor*, que es el suministrado por Sybase y el editor establecido por defecto.

El desarrollador podrá utilizar el editor que sea más de su agrado.

Asimismo, para trabajar en OS/2 se dispone del OS/2 2.1 Presentation manager, numerosos archivos de ayuda y programas de ejemplo, así como del SOMobject's Developer Toolkit.

Esta utilidad permite la creación de aplicaciones que emplean el llamado System Object Model de IBM, un conjunto de normas, utilidades y librerías que se utilizan para la construcción de objetos.

Para trabajar en DOS en modo protegido se suministra el extensor DOS/4GW, el cual se encuentra libre de royalties y tiene soporte de memoria virtual de hasta 32 MB, aunque Watcom también permite desarrollar y depurar aplicaciones basadas en otros extensores, como podrían ser DOS/4G y TNT DOS.

Como último punto que resaltar, se agradece que el CD-ROM esté repleto de ejemplos que ilustran el funcionamiento del nuevo entorno de desarrollo, así como de archivos de ayuda que describen con gran amplitud el funcionamiento de todos los componentes de la herramienta.

Entorno integrado de desarrollo

El IDE de Watcom C/C++ permite la utilización y manipulación de los archivos y herramientas requeridos en el desarrollo de un proyecto. Para conseguir tomar un primer contacto con las mismas, con el paquete son suministrados multitud de ejemplos prácticos para las plataformas de desarrollo Windows y OS/2.

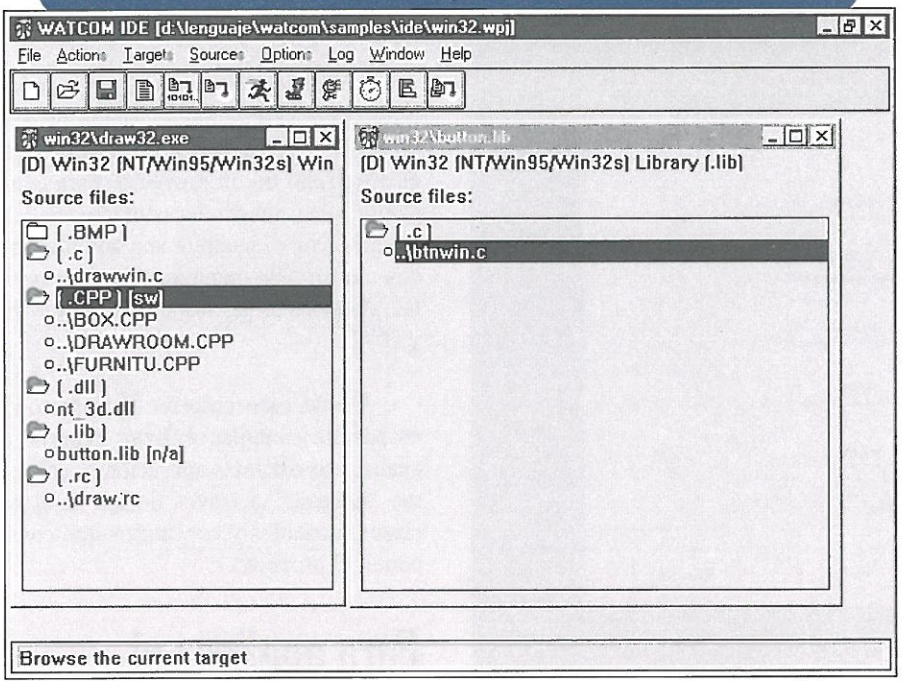
Desde este entorno de desarrollo es posible compilar, enlazar, depurar y analizar las distintas aplicaciones, así como "navegar" a través de los objetos, clases, variables y constantes que componen el proyecto.

Para analizar el rendimiento de las distintas partes de la aplicación se dispone del Watcom Execution Sampler y del Watcom Execution Profiler

El programa depurador posee una interfaz que es configurable por el usuario, proporciona la posibilidad de situar puntos de ruptura en funciones anidadas, tiene capacidad para depurar DLL y permite la ejecución en sentido inverso. Todas estas opciones están disponibles sin perder, en ningún caso, su capacidad multiplataforma.

El Browser permite visualizar las jerarquías de objetos presentes en la aplicación, las funciones, variables y constantes.

Figura 5.



Con el objeto de poder analizar el rendimiento de las distintas partes que componen la aplicación, se dispone de las herramientas Watcom Execution Sampler y del Watcom Execution Profiler.

La utilización combinada de ambas herramientas (en primer lugar es necesario ejecutar la aplicación deseada a través del W.E.S., obteniéndose de este modo un archivo con extensión que permite al W.E.P. visualizar después las estadísticas) permite conocer aquellas partes del código que suelen ser más empleadas y, por lo tanto, éstas son susceptibles de ser depuradas y optimizadas a fondo.

Herramientas para Windows

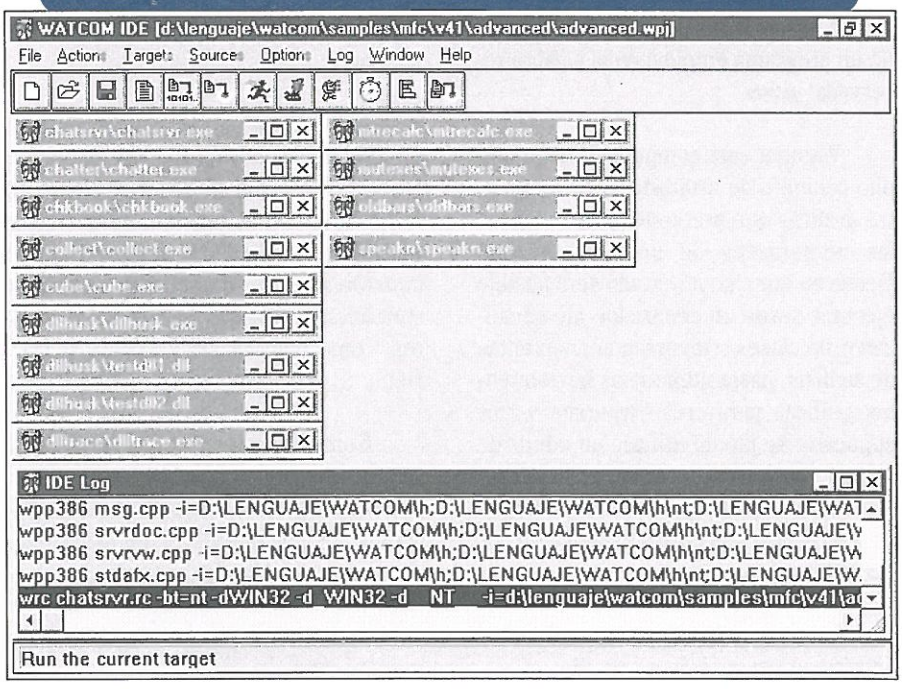
Concebidas exclusivamente para permitir realizar el desarrollo de aplicaciones bajo el sistema Windows, dentro del paquete se dispone de las utilidades siguientes:

- Resource Editors y Resource compiler: como podemos imaginar, permiten la creación y la compilación de recursos.
- Zoom: aumenta una sección de la pantalla.
- Heap Walker: con esta utilidad se puede visualizar la utilización de la memoria. Resulta de mucha utilidad en procesos de depuración de código, que son bastante frecuentes en el desarrollo de aplicaciones.
- Spy: con esta herramienta se permite efectuar la captura de los mensajes que son intercambiados entre una aplicación y el sistema operativo.
- DDESpy: describe la actividad DDE.
- Dr. Watcom: permite conocer el estado del programa y del sistema tras la ejecución de una excepción.

Blue Sky's Visual Programmer

En el momento en que se instala el entorno integrado y las utilidades que lo acompañan, en el grupo de programas que se crea se incluye un elemento que per-

Figura 6.


WATCOM IDE [d:\lenguaje\watcom\samples\mfc\v41\advanced\advanced.wpj]
File Actions Targets Sources Options Log Window Help
IDE Log
wpp386 msg.cpp -i=D:\LENGUAJE\WATCOM\h;D:\LENGUAJE\WATCOM\h\nt;D:\LENGUAJE\WAT
wpp386 srvrdoc.cpp -i=D:\LENGUAJE\WATCOM\h;D:\LENGUAJE\WATCOM\h\nt;D:\LENGUAJE\V
wpp386 srvrvw.cpp -i=D:\LENGUAJE\WATCOM\h;D:\LENGUAJE\WATCOM\h\nt;D:\LENGUAJE\W
wpp386 stdafx.cpp -i=D:\LENGUAJE\WATCOM\h;D:\LENGUAJE\WATCOM\h\nt;D:\LENGUAJE\W.
wrc chatsrvr.rc -bt=nt -dWIN32 -d_WIN32 -d__NT__ -i=d:\lenguaje\watcom\samples\mfc\v41\ad
Run the current target

mite realizar la instalación del *Visual Programmer*.

El Visual Programmer es un entorno de diseño visual para Windows 3.x, Windows 95 y Windows NT

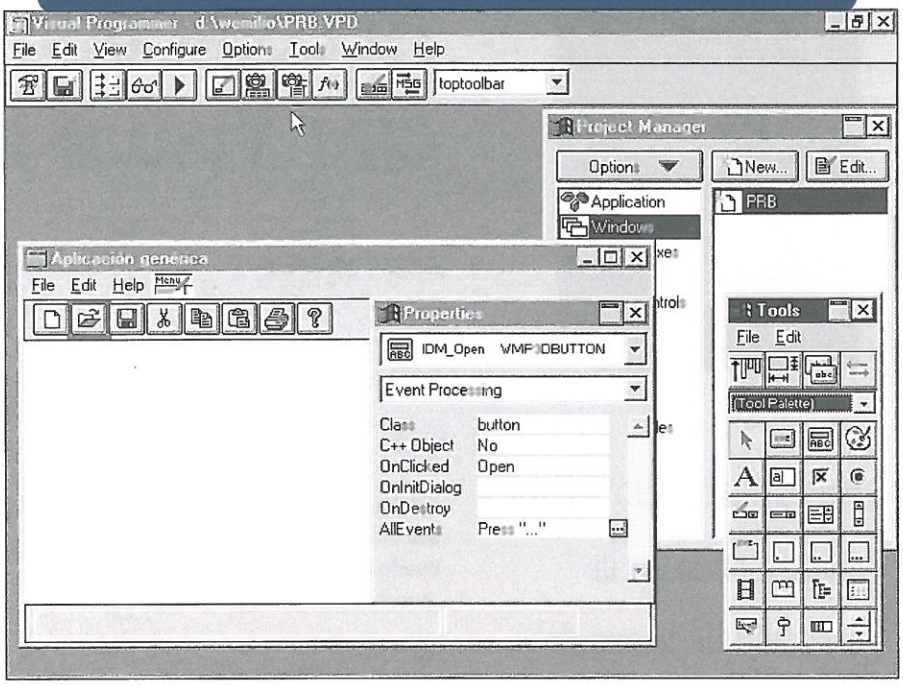
Éste es un entorno de diseño visual para Windows 3.x, Windows 95 y Windows NT que ayuda a construir sofisticadas aplicaciones MFC.

La instalación de este componente puede efectuarse también directamente desde el CD-ROM, accediendo al directorio VP\Disk1 y ejecutando el programa SETUP.EXE.

Con Visual Programmer se puede añadir a una aplicación los elementos habituales de un programa Windows (como pueden ser controles, cajas de diálogo, barras de herramientas, menús desplegables, etc.) de forma instantánea y sin que sea necesario escribir una sola línea de código. Una vez que han sido creados los controles será posible asignar código a sus eventos de forma similar a como es habitual en herramientas del tipo de Visual Basic.

Visual Programmer puede ser invocado desde la opción de menú *Target* del entorno integrado. Una vez que ha sido definido el aspecto final de nuestra aplicación es posible chequear el funcionamiento del programa creado. En el momento en que estemos satisfechos con el resultado de la ejecución de los controles que deseemos, al regresar al IDE, comprobaremos que ha generado un proyecto completo con los controles que hayamos establecido y, si es el caso, con el código que hayamos asociado a los mismos.

Figura 7.



Esta nueva versión de la herramienta es de 32 bits, aunque también es ejecutable en aquellos entornos Windows de 16 bits que dispongan del soporte Win32s (capa de soporte limitado de aplicaciones Windows de 32 bits en el entorno Windows 16 bits). En este caso podrán generarse aplicaciones de 16 bits, aunque el desarrollador deberá tener la precaución de no incluir controles propios de Windows 95. Por lo que respecta a la versión de 16 bits de Visual Programmer, no volverá a ser incluida en Watcom.

Documentación

Al instalar la herramienta, se copian en nuestro sistema una gran cantidad de archivos de ayuda. Estos archivos, que no son la totalidad de los que trae Watcom, constituyen parte del Kit de documentación.

En este formato se encuentran disponibles, entre algunos otros documentos, el manual del usuario, el manual del programador, las guías de referencia de los lenguajes C y C++ y la guía de la librería de clases.

La documentación para DOS está disponible en archivos IHP que deben ser accedidos a través del programa WHELP. Para OS/2 y Windows se tienen las habituales extensiones INF y HLP respectivamente.

Visual Programmer puede ser invocado desde la opción de menú Target del entorno integrado

Esta documentación puede ser solicitada a Sybase en formato impreso, incluyéndose en este caso el libro de Biarne Stroustrup *The C++ Programmming Language*. La documentación está escrita en inglés, excepto las ayudas del sistema operativo Windows, que se pueden encontrar en diez idiomas, incluido el castellano.

VISUAL BASIC

Accediendo a Access con Visual Basic

Jorge Figueroa

En este artículo vamos a adentrarnos en la creación de bases de datos de tipo Access mediante la programación en Visual Basic.

Cualquier lenguaje de programación que sea capaz de manipular bases de datos se puede dividir, habitualmente, en las dos siguientes partes:

- Lenguaje de definición de datos (DDL).
- Lenguaje de manipulación de datos (DML).

Hasta este momento, en los anteriores artículos de esta revista que han sido dedicados a este imprescindible tema, hemos podido tratar los diferentes objetos de acceso a datos, además de los más importantes conceptos dentro de la programación de bases de datos.

DAO nos proporciona el máximo control sobre la creación de bases de datos

Dichos contenidos podríamos encontrarlos enmarcados dentro del conocido como Lenguaje de manipulación de datos (DDL). En las páginas de este artículo vamos a tratar más a fondo este lenguaje de manipulación de datos que nos puede ser de mucha utilidad para construir una base de datos por medio de la programación.

A continuación vamos a centrarnos en la creación de una base de datos para los objetos de bases de datos o DAO, aunque existen otros posibles caminos para conseguir esto mismo, tales como el administrador de datos que viene incluido dentro de Visual Basic, Microsoft Access, otra aplicación externa como DBASE o incluso Fox Pro.

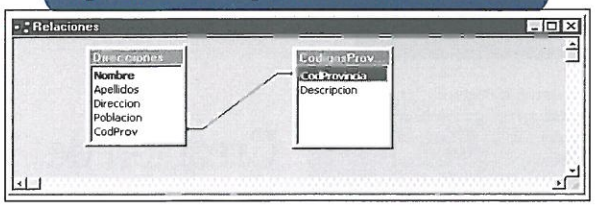
No obstante, DAO nos proporcionará el máximo control sobre la creación de las bases de datos y, además, será la única alternativa que vamos a tener en nuestro proceso de creación, siempre y cuando deseemos conseguir una base de datos que sea capaz de funcionar en tiempo de ejecución.

Creación de una Base de datos

Si lo que queremos es crear una nueva base de datos para el motor de bases de datos *Jet Engine*, en ese caso simplemente tendremos que definir ciertos objetos de acceso a datos que se correspondan con los objetos que deseamos crear, es decir, tablas, campos, índices y relaciones.

Concretamente, para no complicar las cosas de momento, nos proponemos desarrollar una base de datos que sea

Figura 1. Estructura y relación de la base de datos.



muy sencilla. En efecto, tan sólo constará de dos tablas, una de direcciones y otra de códigos de provincia. Dentro de la tabla de direcciones introduciremos los campos siguientes:

- nombre (texto, 20 caracteres)
- apellidos (texto, 20 caracteres)
- dirección (texto, 30 caracteres)
- población (texto, 20 caracteres)
- CodProv (texto, 2 caracteres).

La tabla de Códigos de Provincia contendrá, por su parte, los campos CodProvincia (texto, 2 caracteres) y Descripción (texto, 20 caracteres). La relación es la que se puede observar en la figura 1.

Vamos a crear nuevas variables de tipo objeto para cada objeto de la base de datos que necesitemos, así en una primera rutina llamada "CrearNuevaBD" definimos el objeto base de datos con el que trabajaremos:

```
Dim Db As Database
```

Y acto seguido asociamos esta variable que acabamos de crear a una nueva base de datos:

```
Set Db = CreateDatabase(Dbname,
                        dbLangGeneral)
```

Ahora que tenemos la base de datos creada, podemos ir al siguiente paso y crear las tablas.

Dicha operación la realizaremos en dos rutinas: "CrearTablaCodigos" y "CrearTablaDirecciones" a las que pasaremos como parámetro la variable que es la base de datos. Una vez inmersos en la creación de dichas rutinas, lo primero que necesitaremos es declarar las variables de los objetos que vamos a crear (menos el objeto base de datos que ya lo hemos creado y lo pasamos como argumento). Así crearemos una nueva tabla que contendrá la información de las de direcciones y dos matrices que utilizaremos para poner los valores de los campos y de los índices:

```
Dim Direccion As New TableDef
Dim Campos() As New Field
Dim Indices() As New Index
```

Creación de tablas

Para crear objetos de tipo tabla, campos e índices respectivamente. Una vez en este punto, ya podemos desarrollar el código que nos creará las tablas dentro de la base de datos. En primer lugar redimensionamos las matrices de los índices y de los campos (tendremos cinco campos y dos índices):

```
ReDim Campos(1 To 5), Indices(1 To 2)
```

y acto seguido le asignamos un nombre a la tabla:

```
Direccion.Name = "Direcciones"
```

En este momento estamos preparados para decidir cuáles van a ser los campos de la tabla, es decir, podemos proceder a crearlos. Para ello rellenaremos la matriz de campos que hemos creado anteriormente (Dim Campos() As New Field) con los valores que deseemos y por último los añadimos a la tabla Dirección mediante el método:

```
Append ("Direccion.Fields.Append Campos(I)"):

' Crear los campos
For I = 1 To 5 ' Set properties for fields.
  Campos(I).Name = Choose(I, "Nombre",
                          "Apellidos",
                          "Direccion",
                          "Poblacion",
                          "CodProv")
  Campos(I).Type = Choose(I, dbText,
                          dbText,
                          dbText,
                          dbText,
                          dbText)
  Campos(I).Size = Choose(I, 15, 20, 30, 20, 2)
  Direccion.Fields.Append Campos(I)
Next I
```

En la creación de los campos hemos utilizado algunas propiedades obvias de éstos, tales como el tipo (type), el tamaño (size) y el nombre (name). Los objetos de tipo Field (campos) tienen algunas otras propiedades que nos pueden ser de mucha utilidad en determinados casos, como son las siguientes.

Tabla 1. Valores de la propiedad type del objeto Field.

Valor	Constante	Descripción/tipo de datos
8	dbDate	Fecha/Hora
10	dbText	Texto
12	dbMemo	Memo
1	dbBoolean	Yes/No
3	dbInteger	Integer
4	dbLong	Long
5	dbCurrency	Currency
6	dbSingle	Single
7	dbDouble	Double
2	dbByte	Byte
1	dbLongBinary	Long Binary (Objeto OLE)

- *AllowZeroLength*: nos devuelve o establece un valor, que indica si una cadena vacía ("") será un valor válido para la propiedad Value de un objeto Field (siempre que este sea de tipo de datos Texto o Memo)

- *DataUpdatable*: nos devuelve un valor que indica si los datos de un campo representado por un objeto Field son actualizables.

- *DefaultValue*: nos devuelve o establece el valor predeterminado de un objeto Field. Si el objeto Field no ha sido agregado a la colección Fields, esta propiedad es de lectura/escritura.

- *OrdinalPosition*: nos devuelve o establece la posición relativa de un objeto Field dentro de una colección Fields a la que se agrega. En un objeto no agregado a la colección Fields, esta propiedad es de lectura/escritura.

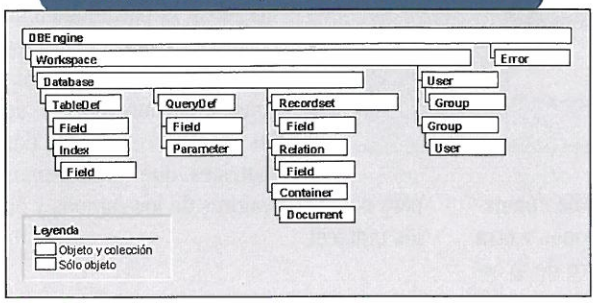
- *Required*: devuelve o establece un valor que indica si objetos Field o Index requieren valores no Nulos. En un objeto no agregado a una colección, esta propiedad es de lectura/escritura.

- *Type*: devuelve o establece un valor que indica el tipo operativo o el tipo de datos de un objeto. En objetos Field, el valor de esta propiedad es de lectura/escritura hasta que el objeto se agrega a una colección o a otro objeto, después de lo cual es de sólo lectura.

- ValidationRule : devuelve o establece un valor que valida los datos de un campo en el momento en que se cambian o se agregan a una tabla. En objetos no agregados a una colección Fields, esta propiedad es de lectura/escritura.

  Esta lista no es exhaustiva por lo que remitimos a la ayuda de Visual Basic para conocer todas las propiedades de los objetos Field.

Figura 2. Estructura de los objetos de acceso a datos. Podemos observar los objetos TableDef, Relation, Field, Index de los que hablamos en el artículo.



Creación de índices

Vamos ahora a crear los índices. De momento, como primer paso, rellenaremos la información de la matriz de índices.

```
Indices(1).Name = "PrimaryKey"
Indices(1).Fields = "Nombre"
Indices(1).Primary = True
Indices(1).Unique = True
Indices(2).Name = "Secundario"
Indices(2).Fields = "Apellidos"
```

Donde *Name* es el nombre del índice, *Fields* son los campos de que consta éste, *primary* nos dice si es el índice primario o no y *unique* nos dice cuándo un índice permite registros con valores duplicados en el campo clave. Dicha propiedad es de solo lectura si el objeto ya ha sido añadido a la colección de TableDef.

Una vez tenemos la información correspondiente a los índices en la matriz de éstos, hemos de vincular los elementos de la matriz a la tabla, para que esta reconozca los índices como suyos. Lo haremos con el método Append.

```
For I = 1 To 2
  Direccion.Indexes.Append Indices(I)
Next I
```

Luego añadiremos la tabla a la colección de tablas para que pase a formar parte de la base de datos:

```
D.TableDefs.Append Direccion
```

Y haremos lo mismo con la otra tabla.

Creación de Relaciones

También nos puede interesar en una base de datos crear una relación entre diferentes tablas, según los valores de las claves principales y ajenas que sean comunes. Sin embargo, para que la relación sea realmente útil, nos será de suma importancia mantener la integridad referencial en el momento en que se añadan o se borren registros. Para hacer todo esto debemos ante todo definir un objeto *Relation*:

```
Dim Relacion As Relation
```

Para que una relación sea realmente útil, nos será de suma importancia mantener la integridad referencial en el momento en que se añadan o se borren registros

Luego creamos el objeto *relation* mediante el método *CreateRealation* del objeto *Database:*

```
Set Relacion = Db.CreateRelation("UnoAVariosForzada")
```

y establecemos las propiedades de las tablas que deseamos relacionar:

```
Relacion.Table = "CodigosProvincia"
    Relacion.ForeignTable = "Direcciones"
```

Luego crearemos un campo para definir el campo de clave principal y ajena común dentro de la relación, usando para ello el método *CreateField* del objeto *Relation*.

```
Dim TmpFields As Field
Set TmpFields = Relacion.CreateField("CodProv")
TmpFields.ForeignName = "CodProv"
```

Por último utilizaremos el método *Append* para agregar el objeto *Field* a su *Relation* y el objeto *Relation* al objeto *Database*.

```
Relacion.Fields.Append TmpFields
    Db.Relations.Append Relacion
```

Y de esta forma habremos creado una relación.

Eliminación de una tabla

En el proceso de modificación de una base de datos existen varias operaciones que se pueden realizar. Una de ellas es la eliminación de una tabla.

Para proceder a eliminar una tabla utilizaremos el método *Delete* de la colección TableDefs. Cuando eliminemos dicha Tabla suprimiremos a la vez todos los campos, índices y datos contenidos dentro de la tabla, por lo que debemos tener sumo cuidado al utilizar esta instrucción. Por ejemplo, para borrar la tabla de CodigosProvincia se procedería como sigue:

```
Db.TableDefs.Delete "CodigosProvincia"
```

Para eliminar una tabla utilizaremos el método Delete de la colección TableDefs

Agregar un campo a una tabla

Podemos agregar un campo a una tabla si lo añadimos a una colección Fields ya existente. El código sería:

```
Private Sub AgregarCampo_Click()

Dim Tabla As TableDef
Dim Campo As Field

Set Tabla = Db.TableDef("Direcciones")
Set Campo = Tabla.CreateField("Prueba",
                    dbText, 20)
Tabla.Fields.Append Campo

End Sub
```

Agregar un índice

Para agregar un índice a una tabla utilizaremos el método *Append* sobre una colección de "Indexes"existente.

```
Private Sub AgregarIndice_Click()
Dim Tabla As TableDef
Dim NuevoIndice As Index
Dim NuevoCampo As Field
Set Tabla = Db.TableDefs("Direcciones")
Set NuevoIndice =
    Tabla.CreateIndex("Poblacion_Indice")
NuevoIndice.Unique = False
Set NuevoCampo =
    NuevoIndice.CreateField("Poblacion")
NuevoIndice.Fields.Append NuevoCampo
Tabla.Indexes.Append NuevoIndice
End Sub
```

Eliminación de un índice

Eliminaremos un índice de manera similar a como eliminamos una determinada tabla, utilizando el método *Delete*. Cabe decir que no podremos eliminar un índice si éste forma parte de una relación.

La sintaxis para eliminar el índice sería la siguiente:

```
Base_de_Datos.TableDefs(tabla).Indexes.Delete
    Índice
```

Eliminaremos un índice de manera semejante a como eliminamos una determinada tabla, utilizando el método Delete

Agregar una tabla

Para agregar una tabla a una base de datos únicamente hemos de añadir un nuevo objeto TableDef a una colección TableDefs ya existente. Por ejemplo en el siguiente código creamos un objeto TableDef (NuevaTabla) y le damos valor con el método CreateTableDef del objeto Database. Hacemos algo similar con el objeto campo. Y una vez hemos dado este paso unimos el campo a la tabla, y la tabla a la base de datos.

```
Private Sub AgregarTabla_Click()
Dim NuevaTabla As TableDef
Dim nuevoCampo As Field
Set NuevaTabla = Db.CreateTableDef("Nueva
    Tabla")
Set nuevoCampo = NuevaTabla.CreatField("Nuevo
    Campo", dbText, 10)
NuevaTabla.Fields.Append nuevoCampo
Db.TableDefs.Append NuevaTabla
End Sub
```

Hemos visto hasta aquí los pasos necesarios, para que nuestra aplicación, mediante la programación, pueda construir bases de datos de tipo Access, o modificar bases de datos ya existentes.

PROGRAMACIÓN DE JUEGOS

Algoritmos para programas de ajedrez

Eugenio Castillo Jiménez

El primer autómata reconocido capaz de jugar al ajedrez (al menos un final de torre-rey contra rey) fue construido en 1890 en la Escuela Técnica Superior de Ingenieros de Caminos de la Universidad Politécnica de Madrid. Su autor fue Torres Quevedo y la máquina era un ingenio mecánico capaz de avisar el jaque y anunciar mate pero estaba limitada a un sólo tipo de posiciones.

Los primeros análisis teóricos para programas de ordenador datan de finales de los cuarenta y los realizaron el inglés Alan Turing y el americano Claude E. Shanon de los laboratorios Bell. No fue hasta 1951, en la Universidad de Manchester, cuando se realizó el primer programa (MADM) capaz de solucionar posiciones de mate complejas, la computadora empleada tenía 16Kb de Ram y su autor fue el alemán Dietrich Günther. Casi por las mismas fechas los americanos Ulam y Stein desarrollaron el MANIAC I que podía jugar partidas completas en un tablero reducido.

Los progresos realizados hasta la fecha tanto a nivel de software como de hardware han sido muy importantes, en la actualidad el módico precio de los ordenadores y la divulgación de determinadas fuentes han dado la posibilidad de celebrar competiciones entre programas de ajedrez incluyendo tanto a programadores profesionales como a amateurs. Ello ha producido un creciente interés en el mundo de la programación aplicada hacia los considerados "juegos inteligentes" como el ajedrez, este campo de investigación-diversión se encuentra aún muy abierto a nuevas ideas y cada año surgen algoritmos que se encargar de suplir los vacíos de juego de los programas. A continuación estudiaremos algunos de estos sistemas.

Cada vez que se intente informatizar un juego se ha de proceder en primer lugar a analizar las reglas que lo componen, si este análisis nos permite crear un programa que sea capaz de jugar legalmente, entonces hemos de conseguir que intente hacer lo posible por ganar. Los factores básicos más importantes a determinar en un análisis inicial son:

1. cantidad de bandos en juego.
2. modo de representación del tiempo (generalmente para controlar las tomas de decisiones).
3. posibilidad de decisiones en dicho tiempo (cantidad de decisiones y bandos que las pueden tomar).

En el caso concreto del ajedrez existen dos bandos (hay juegos en los que se trata únicamente de resolver un problema y "sólo" existe un bando), el tiempo se representa a modo de turnos (jugadas, existen juegos en los que los turnos no existen y las decisiones por bando se toman en tiempo real o sea "en cualquier momento") y en cada turno un bando escoge obligatoriamente una sola de sus posibles decisiones/movimientos (en otros juegos se pueden llegar tomar todas las posibles decisiones, a su vez hay juegos de este tipo en los que el orden de estas decisiones puede dar lugar diferentes posiciones finales). El programador de ajedrez Ken Thompson de los laboratorios Bell definió el ajedrez como "un juego dos-personas suma-cero todo-conocimiento".

A partir de este pequeño análisis previo inicial realizado ya nos podemos

encontrar en condiciones para discutir determinados algoritmos de búsqueda de soluciones.

En primer lugar, lo que tendremos que hacer es representar una búsqueda como un árbol de posibilidades, donde cada nodo de dicho árbol constituirá una jugada (decisión) ejecutada por un bando o equipo (las jugadas pertenecientes a un mismo turno y bando se muestran al mismo nivel dentro del árbol).

Se representa una búsqueda como un árbol de posibilidades donde cada nodo de dicho árbol constituye una jugada del bando o equipo

Cada nodo de dicho árbol puede ser un nodo terminal (un nodo sin hijos). A cada nodo terminal se le asigna un valor, el cual vendrá normalmente determinado por una evaluación en la que se consideran toda una serie de criterios distintos como el equilibrio material, ventajas de espacio, estructuras de peones, y ataques al rey.

Minimax

El algoritmo que selecciona la jugada óptima (máxima) se denomina minimax (fue publicado por primera vez en 1928 y sus autores fueron los matemáticos americanos Von Neuman y Morgenstern), y responde al siguiente principio: de cada nodo con descendencia (o sea no teminal) se escoge el nodo hijo con mayor valor para el bando que mueve. Anotando este valor en cada nodo y continuando el proceso hasta completar todos los nodos obtendremos en el nodo raíz la jugada óptima. Obviamente se pueden dar más de una jugada "máxima" en un mismo nodo, si éste fuera el caso se escoge la primera o mediante un sistema aleatorio se escoge una cualquiera de ellas, puesto que según lo previsto son de fuerza equivalente.

En la tabla 1 podemos observar que los nodos no terminales también poseen valores anotados; éstos se corresponden con el valor considerado como máximo de su nivel inmediatamente superior.

Se denomina **ply** (capa,pliegue) al nivel de profundidad correspondiente a un nodo. En el ejemplo anterior el árbol alcanza dos plys de profundidad (sólo se han analizado dos movimientos seguidos como máximo). Ello implica que si hubiera una combinación de jugadas ganadora que estuviera por encima de ply 2 nuestro programa no la vería, para solucionar esto tendríamos que ampliar la profundidad de análisis de nuestro árbol incorporándole más capas de análisis con lo que el número final de nodos a analizar se vería incrementado de una forma exponencial.

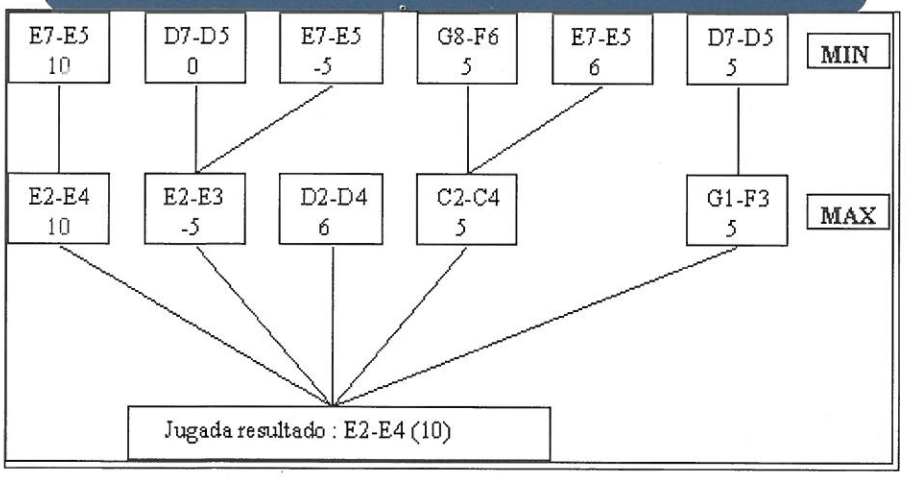
Podemos llegar de este modo a la conclusión de que un análisis a mayor profundidad nos garantizaría un análisis más exacto de una posición, y que un análisis a distintas profundidades nos podría dar respuestas diferentes.

La generación y evaluación de cada nodo conlleva un consumo de tiempo. Ello implica que un análisis a mayor profundidad equivale a un mayor consumo de tiempo. Esto explica la razón por la que un programa de ajedrez que permite más tiempo para pensar o que se ejecuta a mayor velocidad proporciona respuestas más fuertes y distintas a las que daría en otras circunstancias.

Cabe destacar la siguiente cuestión, cuando un programa analiza un árbol no es necesario generarlo en su totalidad y evaluarlo después, de hecho el programa en cada momento no retiene en memoria más de dos nodos por ply. Podemos decir que evaluamos mientras que generamos el árbol, de este modo todo lo ya evaluado no ha de permanecer en memoria.

Veamos un ejemplo en pseudo código de una búsqueda "arbórea" de profundidad 3:

Tabla 1




```
struct tmov {
  int num;
  int mov [60];
}

void main ()
{
      PVS (3);
}

int evaluapos ();
// evaluación de una posición
void generamovimientos ();
// genera los posibles movimientos
//de una posición ya sean blancos o negros
void hazmov ();
```

```
//modifica el estado del tablero después de
//realizar la jugada en curso
void deshazmov ();
//repone la posición anterior a la jugada en curso
int  PVS (int profundidad)
{
   tmov movimientos;
  int f,mejor,valor;

  if (profundidad == 0)
  {
        return (evaluapos ());
  }
  mejor = -12000;
  generamovimientos (movimientos);

  for (f = 0;f<movimientos.num;f++)
 {
   hazmov (movimientos.mov [f]);
   valor = -PVS (profundidad - 1);
   if (valor > mejor)
     mejor = valor;
 }
 return (mejor);
}
```

Como se puede comprobar, **PVS** es una rutina recursiva que genera y evalúa el árbol. **valor** toma el valor retornado por PVS cambiado de signo, se realiza este cambio para representar la diferencia entre un turno max y otro min (lo que para el rival es positivo para el propio bando es negativo).

Alfa-beta

Casi al mismo tiempo que se implementó el minimax (para jugar al ajedrez) se descubrió el **alfa-beta** (Shanon es el primero que hace referencia a éste) que es una simplificación del primero. El nuevo algoritmo se basa en lo siguiente, digamos que las blancas en ply (1) han conseguido en la primera jugada un valor de diez, al comenzar a analizar la segunda jugada blanca ply (1) las negras han obtenido un valor cinco en su primera jugada del ply (2) (que retornaría hacia ply (1) con valor -5) , cualquier otra jugada de valor inferior a diez ya no mejora el ply (1) luego el análisis de las restantes jugadas negras de ply (2) carece de objeto puesto que no cambiarán el valor blanco de ply (1), por ello se retorna de un modo automático el valor cinco obtenido y nos ahorramos el resto de la evaluación negra.

En la tabla 2 se puede observar un ejemplo ilustrado, las zonas sombreadas muestran el ahorro.

Podemos comprobar que esta simplificación carecería de objeto si el algoritmo de búsqueda primero generase todo el árbol y luego lo evaluase. Veamos cómo se habría de modificar el código anterior (en main () y PVS ()) para implementar el **alfa-beta**. La principal diferencia se encuentra en lo que sucede después de "if (valor > mejor)".

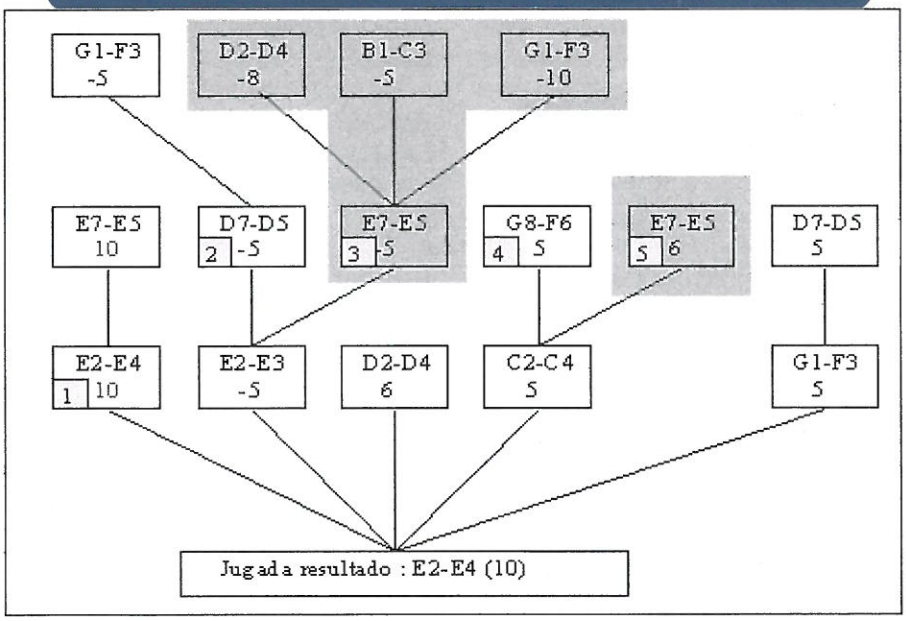
```
void main ()
{ PVS (3,-12000,12000);
}

int  PVS (int profundidad, int alfa, int beta)
{ tmov movimientos;
  int f,mejor,valor;

  if (profundidad == 0)
  {
      return (evaluapos ());
  }

   mejor = -12000;
  generamovimientos (movimientos);

  for (f = 0;f<movimientos.num;f++)
 {
   hazmov (movimientos.mov [f]);
   valor = -PVS (profundidad - 1,-beta,-alfa);
   if (valor > mejor)
   {
       if (valor  > beta)
         return (valor);
       mejor = valor;
       if (mejor > alfa)
         alfa = mejor;
   }
 }
 return (mejor);
}
```

El nombre de alfa-beta proviene de las dos variables que determinan la ventana de búsqueda (llamamos ventana al margen superior e inferior de los valores que pueden optar a ser jugadas resultado).

Existe otro detalle de vital importancia en el algoritmo de simplificación del alfa-beta: cuanto antes se analice la jugada principal (o una de valor aproximado) antes se conocerá el alfa-beta óptimo y mayor será el ahorro de nodos producido.

Ello ha conducido a que los programadores se lancen a buscar sofisticados sistemas de búsqueda que puedan encontrar cuanto antes una jugada máxima que

Tabla 2



reduzca de inmediato la ventana alfa-beta. Un sistema básico y reconocido como útil en la práctica consiste en realizar la búsqueda en el árbol mediante profundidades incrementales. A cada análisis realizado obtendremos una variante principal que aunque no sea la variante final sí será en un alto grado de certeza al menos aproximada. De este modo el siguiente análisis a profundidad + 1 se comenzará analizando en primer lugar la variante obtenido en la anterior búsqueda como la mejor. De nuevo modificamos el código y obtenemos lo siguiente:

```
int tiempo ();
//tiempo restante de búsqueda permitido, equivale
      // a un nivel de juego si hemos sobrepasado
      // el límite no seguimos analizando

void main ()
{ int profnominal = 0;

  while (tiempo)
  { profnominal++;
    PVS (profnominal,-12000,12000);
  }
}
```

Se ha venido a demostrar que en la mayor parte de los casos esta mejora produce excelentes resultados, incluso el incremento de profnominal (profundidad nominal) en un sólo ply (capa) es mejor que en dos o más (al menos en la mayor parte de los casos).

Valores hipotéticos

Existen otras formas sofisticadas de reducir el árbol a generar y consisten, fundamentalmente, en introducir valores previos e hipotéticos en la ventana alfa-beta.

Supongamos que en ply (3) hemos obtenido como mejor jugada Cf1-g3 con valor +0.3, si ahora vamos a analizar con ply (4) existe una alta probabilidad de que de nuevo Cf1-g3 sea la buena con un valor +-0.6 de oscilación.

Teniendo como base esta suposición cambiamos de nuevo el código.

```
void main ()
{ int profnominal = 0;
  int mejor = evaluapos();
  int alfa=mejor - 60,beta =mejor + 60,

  while (tiempo)
  { profnominal++;
    mejor = PVS (profnominal,alfa,beta);

    if (mejor < alfa)
    {
       mejor = PVS (profnominal,-12000,mejor);
    }

    if (mejor > beta)
    {
       mejor = PVS (profnominal,mejor,12000);
    }

    alfa = mejor - 60;
    beta = mejor + 60;
  }
}
```

Cada vez que obtengamos un valor mediante una ventana alfa-beta hipotética pueden ocurrir tres cosas:

1. El resultado está comprendido en la ventana, no hay ningún problema y el valor obtenido es fiable.

2. El resultado está por debajo del alfa (salto por alfa), lo que significa que la suposición ha pecado de optimista y el contrario tenía algo mejor de lo previsto, el resultado real puede estar aún más por debajo, es preciso reevaluar con la misma profundidad nominal sin límite inferior alfa, y con límite beta igual al resultado obtenido.

3. El resultado está por encima del beta previsto (salto por beta), la posición es aún mejor de lo esperado, es preciso reevaluar con alfa con el valor "mejor" y un beta sin límite (osea 12000).

En cualquier proceso de análisis que sigamos el resultado final debe de ser el mismo, resulta evidente que con una ventana alfa-beta aún más cerrada podemos acelerar la búsqueda, pero también corremos el peligro de producir demasiados saltos por alfa o por beta. Ya es cuestión del tino de cada programador encontrar la ventana adecuada a sus algoritmos.

Evaluación estática

Basándonos en el pograma TPChess veamos uno de los sistemas de evaluación más simples que se conocen el de la **evaluación estática**, supongamos que definimos un arrray de 64 enteros (uno por casilla) para cada uno de los seis tipos de piezas que existen, teniendo en cuenta también su diferencia de color tendríamos doce tableros de 64 enteros. En cada uno de dichos enteros colocamos el valor que suponemos tiene esa casilla para esa pieza en la posición inicial de la búsqueda.

Cada vez que efectuamos un movimiento, restamos el valor de la casilla de partida y sumamos el de la casilla de destino. De esta forma habremos obtenido una evaluación aproximada del valor posicional de la pieza. Por lo general a cada pieza se le da un valor material que puede ser el siguiente (Peón : 100, Caballo : 300; Alfil : 315; Torre 450; Dama 850; Rey 600). Existen programas que cambian el valor material de las piezas en los finales, por

Tabla 3. Ejemplo de tabla de valores para torres blancas.

9	9	11	10	11	9	9	9
4	6	7	9	9	7	6	4
9	10	10	11	11	10	10	9
8	8	8	9	9	8	8	8
6	6	5	6	6	5	6	6
4	5	5	5	5	5	5	4
3	4	4	6	6	4	4	3
0	0	0	0	0	0	0	0

ejemplo los peones aumentan su valor. Este valor material de todas las piezas sumado al posicional dado por las tablas constituye el valor de una posición, y que es en sí el único criterio de selección.

El problema de este método de evaluación es que no se tienen en cuenta las circunstancias cambiantes de una combinación

El problema real de este método de evaluación reside en el hecho de que no se tienen en cuenta las circunstancias cambiantes que se darán durante el transcurso de una combinación o de unos simples movimientos y que pueden dar lugar a graves equívocos posicionales. Precisamente de aquí el nombre de evaluación estática.

Este sistema se utilizó mucho en la era anterior a los 80386. A partir de su aparición, la velocidad de los procesadores ha permitido que los programadores se lucieran con sofisticados sistemas que elevaron los programas a los niveles de jugadores profesionales humanos.

Búsqueda de tranquilidad

Hasta ahora sólo hemos visto árboles que no expanden su análisis más allá de una determinada profundidad, ya comentamos con anterioridad que ello podía conllevar el que no se vieran combinaciones que continuasen por encima del árbol. Es pues natural que se hallan creado algoritmos que paliasen este defecto, uno de estos es el de la **búsqueda de tranquilidad** (o quiescencia) también llamado por otros programadores **búsqueda de capturas.**

Pongamos el caso que hemos llegado a un nodo al límite de profundidad nominal (profundidad de análisis establecida) es posible que el contrario con un simple movimiento de captura nos gane una pieza haciendo inválida toda la evaluación del nodo terminal. Otra posibilidad es que el movimiento final fuera una captura y que el contrario tuviera una recaptura, la cual es preciso calcular.

En definitiva si el nodo terminal es significativo (osea que su valor puede llegar a ser máximo, por lo que con toda seguridad su valor supere al alfa) es conveniente realizar un análisis de capturas y recapturas. Esta pequeña ampliación del análisis fuera de la profundidad nominal establecida nos obliga a adaptar el PVS de la siguiente forma:

```
int  PVS (int profundidad, int alfa, int beta)
{ tmov movimientos;
  int f,mejor,valor;

  if (profundidad == 0)
  {
    mejor = evaluapos ();
    if (mejor > beta)
      return (mejor);
      //sin hacer nada ya tenemos
      // mejor evaluación
    if (mejor > alfa)
      alfa = mejor;
    generacapturas (movimientos);
  }
  else
  {
    mejor = -12000;
    generamovimientos (movimientos);
   }

  for (f = 0;f<movimientos.num;f++)
  {
    hazmov (movimientos.mov [f]);
    valor = -PVS ((profundidad) ? profundidad - 1 :
       0,-beta,-alfa);
    if (valor > mejor)
    {
      if (valor > beta)
        return (valor);
      mejor = valor;
      if (mejor > alfa)
        alfa = mejor;
    }
  }
  return (mejor);
}
```

El PVS con búsqueda de tranquilidad cambia ante todo a partir de la cuarta línea. Aquí es identificado el análisis fuera de profundidad nominal (profundidad == 0) y se procede a dar como valor inicial del nodo (sin efectuar análisis de árbol alguno) a la evaluación posicional realizada. Si ésta fuera superior al beta se salta hacia fuera, de lo contrario se actualiza el alfa si fuera necesario.

Esta es la principal diferencia respecto a los nodos que se encuentran dentro de profundidad nominal. Otro detalle a considerar es que no se generan todos los movimientos sino sólo los referentes a capturas. No he tenido en cuenta que se pueden producir casos de jaque para hacer más sencilla la exposición de la idea.

Con este método no se generan todos los movimientos sino sólo los referentes a capturas

Si nos fijamos con atención, ahora la longitud de las ramas del árbol van a ser completamente diferentes en la mayor parte de los casos.

El sistema de búsqueda de tranquilidad descrito es el más común de los usados en los programas públicos hasta la fecha. El incremento de la fuerza de juego es notoria, aunque desde un punto de vista ajedrecístico sigue siendo insuficiente como para poder ser efectivo de por sí contra jugadores humanos que posean unas buenas bases de juego.

Variante Principal y Alfa-beta

Cada vez que se realiza el análisis de un árbol, éste nos dá una respuesta, a la que se denomina **variante principal** (VP,PV en inglés). En algunos programas existe un aprovechamiento de la ventana alfa-beta. Una vez que hemos terminado de analizar la VP y si el valor obtenido se encuentra dentro de los límites del alfa-beta previsto, entonces se reduce la ventana alfa-beta al mínimo posible.

Veamos en un ejemplo de cómo quedaría la sección de PVS que realiza la llamada al siguiente ply:

```
//VP indica si estamos o no en una jugada de
// la variante principal

for (f = 0;f<movimientos.num;f++)
{
  hazmov (movimientos.mov [f]);
  if (!VP)
  {
      valor = -PVS ((profundidad) ? profundidad -
          1 : 0,-alfa -1,-alfa);

    if (valor > alfa && valor < beta)
      valor  = = -PVS ((profundidad) ?
          profundidad - 1 : 0,-beta,-valor);
  }
  else
    valor  = -PVS ((profundidad) ?
          profundidad - 1 : 0,-beta,-alfa);

  if (valor > mejor)
  {
    if (valor  > beta)
      return (valor);
    mejor = valor;
    if (mejor > alfa)
      alfa = mejor;
  }
}
```

Esta pequeña modificación realizada hace más efectivo el ahorro de nodos (estadísticamente hablando), sin embargo, no todos los programas lo emplean. El CRAFTY es un fiel partidario de esta idea.

Prebúsquedas

Imaginemos que nos encontramos en un análisis a profundidad nominal 6 y estamos en ply 2, tenemos por tanto 4 plys de búsqueda por delante. Si no estamos en una VP no tenemos jugada conocida con garantías por la que jugar.

En estos casos el CRAFTY hace uso de un recurso muy simple pero efectivo, realiza una prebúsqueda a nivel (profundidad - 2) y luego una vez obtenida una jugada principal lanza el análisis a nivel (profundidad) mirando en primer lugar la jugada que anteriormente ha sido dada como mejor. Sólo he visto utilizar esta idea (muy lógica) al CRAFTY, pero me temo que únicamente es efectiva en partidas de torneo de larga duración en las que se disponga de más de dos minutos por jugada para pensar.

Permanent Brain

Se denomina así a la capacidad de los programas de ajedrez para pensar "permanentemente" incluso después de haber movido y estando en el turno de su rival esperando la respuesta de éste. Para hacernos una idea más precisa de la situación hemos de conocer un poco la mecánica de una partida de ajedrez entre dos programas de ordenador.

Generalmente si se trata de partidas oficiales ambos rivales mueven sus piezas sobre el mismo tablero y disponen de un tiempo determinado para pensar (se suele seguir un ritmo de tiempo por jugadas). Cuando un ordenador genera un respuesta el operador humano realiza el movimiento sobre el tablero y pulsa su reloj para que comience a contar el tiempo de su rival, el operador contrario introduce la jugada en su ordenador y espera la respuesta continuando el proceso. De este modo los ordenadores disponen de un tiempo muerto entre jugada y jugada, este tiempo se corresponde con el que emplea su rival para responder.

Si nosotros empleamos este entretiempo en pensar una respuesta a la respuesta entonces podremos contestar de inmediato ahorrando tiempo, o pensando aún más la jugada que nos corresponde.

Este hecho es aún más efectivo cuando la jugada que realiza el contrario ha sido la que el ordenador predijo en su momento. La estrategia (que muchos humanos desconocen) de hacer jugadas diferentes a las previstas por el ordenador aumentan nuestras posibilidades contra estos "seres".

El algoritmo de búsqueda del turno nulo inserta nodos sin movimientos en el árbol de búsqueda

El tener o no tener el "permanent brain" es un factor decisivo en partidas de campeonato entre programas de la misma fuerza.

Turno nulo

La referencia más conocida actualmente acerca del turno nulo la publicó el austríaco Christian Dönninger en Septiembre de 1993 en la revista ICCA. Como su propia palabra indica, el algoritmo de búsqueda inserta nodos sin movimientos en el árbol de búsqueda; esto es ilegal pero persigue una idea muy humana que sería la siguiente: ¿en qué me amenaza mi rival si no hago nada?

Este turno nulo sólo posee validez dentro de profundidad nominal siempre que se emplee de un modo muy especial. Mostremos la implementación:

```
int  PVS (int profundidad, int alfa, int beta)
{ tmov movimientos;
```

```
int f,mejor,valor,valorinicial;
valorinicial = evaluapos ();

if (profundidad == 0)
{
  mejor = valorincial;
  if (mejor > beta)
    return (mejor);
     //sin hacer nada ya tenemos
     // mejor evaluación
  if (mejor > alfa)
    alfa = mejor;
  generacapturas (movimientos);
}
else
{
   mejor = -12000;

  //****TURNO NULO****
  if (!VP && profundidad > 2 && ...)
  //es preciso que el turno anterior
  // no sea nulo y que no estemos en jaque
  {
     valor =  -PVS (profundidad - 2 ,-beta,-alfa);
     if (valor > mejor)
     {
        if (valor  > beta)
            return (valor);
        mejor = valor;
        if (mejor > alfa)
            alfa = mejor;
     }
     //Esta parte del código es opcional y
     // se denomina "deep null move"
     if (valor < valorinicial + 150)
        profundidad++;
  }

  generamovimientos (movimientos);
 }

for (f = 0;f<movimientos.num;f++)
.....

return (mejor);
}
```

Vemos que la llamada al PVS se efectúa con un profundidad decrementada en dos en vez de en uno que sería lo normal, esto sirve para amortiguar el efecto de no mover a cambio de eliminar un ply de análisis al rival. Los efectos son contundentes y el programa analiza aproximadamente un ply más en condiciones normales.

Puesto que se trata de una apuesta-aproximación nos encontramos ante un arma de doble filo debido a que, por lo general, casi todas las combinaciones tácticas ganadoras se pueden adivinar antes de lo que sería normal, pero, por contrapartida, cuando uno se encuentra en posición inferior puede llegar a ser contraproducente.

La parte opcional del código (el deep null) produce un zoom del árbol sobre los puntos tácticos más críticos

Como efecto negativo seguro tenemos la pérdida del juego estratégico. Otra gran contrapartida se encuentra en que puede llegar a producir valoraciones claramente engañosas en los finales de partida.

La parte opcional del código (el deep null) produce un zoom del árbol sobre los puntos tácticos más críticos, evidentemente esto redunda en favor del juego táctico.

Casi todos los programas públicos modernos emplean de una forma u otra el turno nulo. En mi opinión el CRAFTY lo hace mejor que ningún otro precisamente por las condiciones que impone para no realizar el turno nulo. Yo personalmente utilizo mi propia versión casera y me considero más bien partidario de utilizarlo sólo en casos muy determinados, de otro modo es siplemente una ruleta rusa espectacular.

Cabe mencionar que Dönninger demostró en su famoso artículo mediante ejemplos de posiciones que programas profesionales de primera fila utilizaban esta idea. Uno de sus máximos exponentes es el programa FRITZ cuya característica más destacada es su juego táctico.

Programas públicos

Existen programas de acceso público que pueden servir como apoyo para consolidar el estudio de los algoritmos a los que se ha hecho referencia en el presente artículo.

En el CD-ROM que acompaña la revista podréis encontrar alguno (si no todos, según la rapidez con que sean encontrados) de los siguientes programas públicos con sus fuentes respectivos:

- **TPChess:** se trata de un programa flojo y muy antiguo (tiene aproximadamente quince años) pero que reune muchas ideas originales y se presta ante todo a ser salvajemente acelerado.

De hecho uno de los programadores españoles (Killer de Valencia) posee el récord nacional de velocidad por nodo con una reversión en ensamblador de este programa que llega a alcanzar los 60.000 nodos por segundo en un entorno de 16 bits.

- **GNU:** su principal autor fue John Stanback, actual programador del Zarkov. Su característica más importante reside en el sistema de evaluación, aunque su generador de movimientos también se caracteriza por ser muy original.

- **CRAFTY:** su programador más asiduo es Bob Hyat. Este programa fue originalmente diseñado para un CRAY y está reconocido como el origen teórico del Deep Blue.

Como característica más destacada del mismo, cabe reseñar la utilización masiva de máscaras de bits para realizar evaluaciones y generar movimientos.

La última versión del CRAFTY obtuvo una buena posición en el pasado mundial de Yakarta.

Ventajas de los Algoritmos Genéticos

Lo más sorprendente y útil de los AAGG es que no es necesario disponer de un conocimiento profundo del problema a resolver, sino que se parte de estructuras simples que interactúan, dejando que sea la evolución quien haga el trabajo. Un AG es un método de programación opuesto al habitual: en vez de descomponer un problema en sub-problemas, creamos los sub-problemas más sencillos que podamos imaginar y dejamos que se combinen entre sí.

Únicamente basta con ser capaces de identificar *cualitativamente* en qué casos las cadenas se acercan o se alejan de la solución buscada, para que el sistema se perfeccione automáticamente.

Un Algoritmo Genético es un modo de programación opuesto al habitual. No se descompone un problema en subproblemas: se hace lo contrario

Otra gran ventaja de los AAGG es que son *adaptativos*: son capaces de solucionar problemas que cambian en el tiempo.

La gran sencillez de los AAGG los hace muy atractivos para los programadores, pero existen casos en los que puede ser difícil, imposible o poco práctico aplicarlos. Se podría hablar del *problema de la variedad*, el *problema de la reproducción* y el *problema de la selección*, para los que se van a proponer algunas soluciones.

El problema de la variedad

Un AG trata de explorar las regiones más prometedoras de un enorme espacio de posibilidades. Al encontrar una zona con pesos altos, ésta se explora más a fondo. Pero hay que evitar que el algoritmo se estanque en una determinada zona, produciendo multitud de cadenas muy parecidas.

El aumento de variedad se consigue con las mutaciones, pero esto no suele bastar. Una solución es seleccionar un gran número de agentes, para evitar que un pequeño grupo repita sus caraterísticas en una gran población. Otra opción que suele tener éxito es seleccionar, además de la población con mayor peso, una fracción de la de menor peso.

También se ha de procurar que, a la hora de la reprodución, los progenitores se tomen al azar y no en orden de pesos, ya que es más probable que frases de pesos parecidos posean palabras parecidas. Otra elección más radical es hacer que si se reproducen dos cadenas iguales, los hijos sean mutaciones en todos sus elementos; o recorrer de vez en cuando toda la población eliminando cadenas idénticas.

También podemos generar *ráfagas de mutaciones*: en vez de establecer una probabilidad de mutación independiente para cada creación de una palabra, podemos hacer que si se produce una mutación en un elemento, el siguiente elemento tenga una probabilidad de mutación mayor. En la naturaleza las mutaciones se pueden producir, entre otras causas, por un exceso de radiación. Es lógico pensar que si no hay exceso para un gen, es menos probable que lo haya para el siguiente, y viceversa.

Pero, en definitiva, lo que se busca es que el cálculo del peso no sea independiente para cada cadena, sino relativo: que se le asigne un peso alto cuando es evaluada positivamente por sí sola, y además no existen cadenas que se le parezcan. Para ello podemos analizar las cadenas de mayor peso que una dada y disminuir el peso de esta cadena si existe un gran parecido con las demás cadenas.

En la vida natural ocurre esto mismo: ¿por qué no somos todos altos, rubios, fuertes y guapos? ¿Por qué hay tántas especies en la naturaleza? Los seres parecidos compiten unos con otros, pero la diversidad y la especialización convive pacíficamente. Todas estas opciones están contempladas en el programa ¿Qué criterio aplicar? Hay que tener en cuenta por una parte que trabajamos con azar, y que una sóla ejecución no es muy significativa, y por otra, que puede haber opciones que "lógicamente" sean mejores, ya que disminuyen el número de ciclos necesarios para llegar al peso máximo, pero que en la práctica no son rentables por el incremento del tiempo de proceso debido a la complejidad de los cálculos.

Encontrar la combinación de parámetros óptima resulta por tanto, costoso. Pero existe una forma de automatizar la elección de estos parámetros. ¿Cuál será? Hablaremos de ello en el siguiente artículo de la serie.

El problema de la reproducción

Las características positivas de los individuos deben poder transmitirse a la descendencia. Esto lo hemos conseguido fácilmente en el caso de las frases, pero en otros problemas no será tán sencillo. Por ejemplo, si estoy buscando la secuencia de acciones con la que se consigue un determinado objetivo (isí, sí, estamos hablando de *programación automática*!), el patrón 234, 523, 12, 765, 256 (¿códigos ensamblador?) podría representar dicha secuencia de acciones, suponiendo que se ha asignado a cada número una acción o instrucción.

suscríbete

a Sólo Programadores

y consigue un magnífico descuento

suscripción normal

ahorro **20%**

12 revistas (1 año) por sólo...

9.350 ptas.

suscripción estudiantes (carreras técnicas)

ahorro **40%**

12 revistas (1 año) por sólo...

7.050 ptas.

21
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAST
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
WWW: EL PROCESAMIENTO DE FORMS
LINUX: EMULADORES DE MS-DOS
Y además:
VISUAL TOOLS: AQUÍ Y AHORA

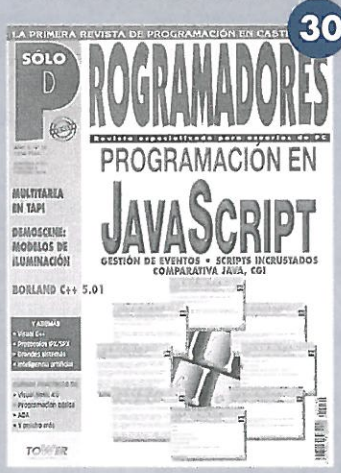
23
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAST
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
EL PODER DE LOS 32 BITS
WATCOM C++
ESTUDIO DE LOS APPLETS DE JAVA
LINUX: CÓMO CREAR DISCOS DE EMERGENCIA A MEDIDA
Y además:

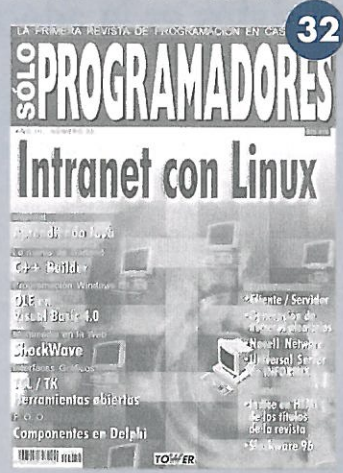
24
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAS
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
ESTUDIO DEL FORMATO WAV
LAS VENTANAS LLEGAN AL WEB
DEPURACIÓN DE CÓDIGO EN VISUAL C++
Y además:
PROGRAMACIÓN DEL PUERTO SERIE

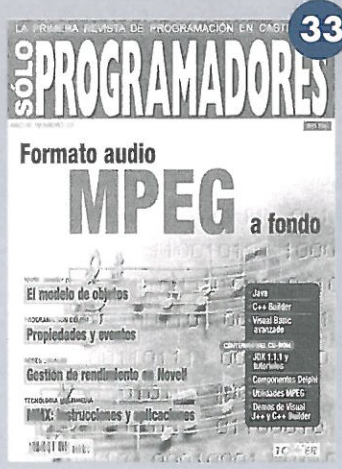
números atrasados

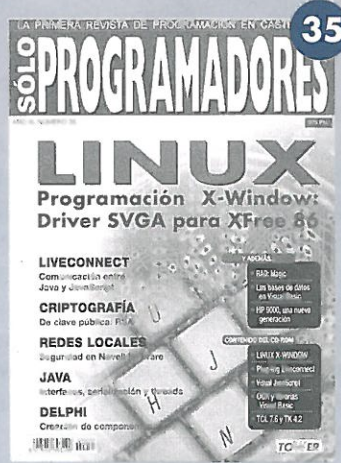
SÓLO PROGRAMADORES

completa ya tu colección









Sin embargo, la probabilidad de que mediante la combinación de dos cadenas válidas de este tipo, se genere una cadena válida es muy baja. ¿Alguien ha cambiado alguna vez una línea de un programa al azar y el programa ha seguido funcionando? Ésta parece una dificultad insalvable, pero no lo es. Una vez más, la vida natural nos da la pista: en realidad, los genes, más que determinar directamente las características que definen un nuevo individuo (color de los ojos, etc.), definen un conjunto de reglas de desarrollo que aplicadas y combinadas provocarán dichas características. Para ello, la dotación genética, que existe en todas las células de cada ser vivo, controla la realización de ciertas reacciones químicas, de manera que los genes son los que determinan, pero no directamente, la constitución y comportamiento de dicho individuo.

Es difícil saber cuándo una estructura de datos es la adecuada

Supongamos que vamos a construir un AG que juegue al **tres en raya**. Debemos decidir qué tipo de información va a estar almacenada en las cadenas protagonistas del algoritmo. Lo más sencillo es que los patrones contengan cada una de las posiciones de las fichas colocadas. Por ejemplo, una serie válida es: (1,1)-(1,2)-(1,3). Esto significa que el individuo representado por esta cadena colcará su primera ficha en la posición (1,1), la segunda en (1,2) y la tercera en (1,3). Es posible que este individuo haya ganado alguna partida con esta secuencia, y por tanto sea seleccionado. Sin embargo, la información almacenada genéticamente no es representativa de cómo y porqué ganó, ya que depende completamente de los movimientos realizados por su contrincante. La repetición de la secuencia de movimientos (1,1)-(1,2)-(1,3) puede originar con facilidad partidas no válidas, al intentar colocar una ficha en una casilla ocupada, y evidentemente la combinanción de dos cadenas con esta estructura raramente producirá una partida válida, mucho menos una cadena vencedora.

La solución al problema de la reproducción es encontrar una estructura de datos capaz de contener la información crítica necesaria para alcanzar el objetivo propuesto. Aquí entramos en el tema de los métodos de representación del conocimiento ámpliamente estudiado en los *sistemas expertos*. En muchos casos, es suficiente un sistema de reglas.

En el **tres en raya**, podríamos tener reglas del tipo *"si observo tal situación, realizo tal acción"*. Cada entidad estaría compuesta por un conjunto de reglas de este tipo. Al jugar una partida, la entidad, observa el estado del tablero, y si posee alguna regla que aplicar, la aplica. En caso contrario, juega al azar.

En este caso, en las entidades sí está almacenado el conocimiento clave para ganar el juego, que son las reglas, no los movimientos. Cada conjunto de reglas será una solución. El elemento mínimo que se combina o muta es una regla completa. Y una posible función de evaluación en este caso es provocar una partida entre dos entidades.

Sistemas no-lineales

¿Entonces, el uso de reglas solucionará siempre el problema de la reproducción? Es difícil saber cuándo una estructura de datos es adecuada; sin embargo, sí existe una forma de saber por anticipado cuándo una estructura de datos **no** es adecuada.

El método es el siguiente: si al modificar un elemento cualquiera de una cadena seleccionada, la cadena resultante es pésima, entonces la estructura de datos elegida no es correcta, ya que los componentes de la cadena por sí sólos no contienen la información que hace que esa cadena sea seleccionada. Expresado de forma estricta: *"Cuando el objetivo de una entidad se puede calcular como una función no-lineal de los componentes de la entidad, la estructura de datos es incorrecta"*.

Una función no-lineal es aquella en la que variaciones muy pequeñas de los valores de sus variables producen variaciones muy grandes en el resultado de la función. El ejemplo por excelencia de problema no-lineal es **la lotería**. El cambio de uno sólo de los dígitos de nuestro billete puede producir cambios enormes en la cantidad que se recibe como premio.

El tres en raya con reglas es bastante lineal: cada regla, o es buena o es mala. Si una entidad seleccionada posee 40 o 50 reglas, probablemente podamos cambiar una sin que se convierta en un pésimo jugador. Pero en casi todos los problemas existe una cierta *no-linealidad* inevitable. Por ejemplo, pueden producirse por azar reglas muy interesantes, pero que no son seleccionadas porque necesitan de otras para demostrar su utilidad y producir un peso alto. Es decir, las *no-linealidades* son trozos de cadena cuya utilidad depende de los valores de otros segmentos, y que sólo son útiles cuando todos ellos contienen determinados datos, desapareciendo la utilidad al fallar uno de ellos.

Cuanta más *no-linealidad* exista, más lentamente convergerá nuestro AG. Esto es completamente inevitable, ya que no es posible encontrar algo si no sabemos qué es lo que estamos buscando. ¡Sin embargo, la naturaleza parece que sí lo consigue! ¿Cómo han podido generarse gradualmente órganos complejos, como los ojos, que no pueden depender de una única mutación, si sólamente el órgano completo es útil al individuo? No siempre es fácil explicar los rasgos que presenta un organismo con un mecanismo adaptativo.

En mi opinión, este planteamiento es erróneo. Los seres vivos sacan beneficio de cualquier órgano a medio formar. El sistema visual de la rana está "sintonizado" solamente a cuatro clases de estímulo, estos son: el contraste de luz y oscuridad; un borde de luz u oscuridad; una repentina disminución de la luminosidad; y el constante movimiento de un pequeño

objeto negro. ¿De verdad únicamente el órgano completo (visión en colores) es útil al individuo? Es habitual considerar nuestra inteligencia o la visión de un halcón como perfectas, en vez de simples estadios intermedios, que llegarán a niveles mucho más avanzados gracias a la evolución. La naturaleza es mucho más lineal de lo que parece, y nosotros podemos ser "la rana".

El método de reproducción

Bien, estamos resolviendo un problema con ciertas *no-linealidades* y por fín hemos conseguido que aparezcan juntos los valores deseados en los segmentos no-lineales. ¿Vamos a dejar que esta información tán valiosa se pierda en la siguiente generación? Si combinamos segmentos de la cadena de cualquier forma, es muy probable que dividamos el trozo no-lineal, perdiéndose todas sus propiedades.

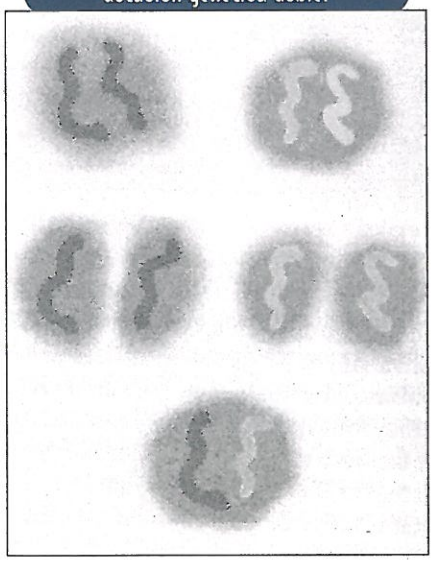
¿Existe alguna forma en la que la naturaleza fuera capaz de evitar esto? Sí. Podemos hacer que segmentos con iguales valores en cadenas que se reproducen vean reforzada su unión, de forma que en toda la descendencia sucesiva aumente la probabilidad de que esos segmentos se hereden completos de un sólo progenitor, sin combinarse con los del otro. Este criterio sería falso cuando se reprodujeran dos cadenas hermanas, ya que aunque tendrían segmentos iguales, esto no sería un indicador de rasgos positivos, sino que únicamente se trata de la parte que ambas han heredado del mismo progenitor. Esta opción presupone que existe un número muy grande de cadenas y que la probabilidad de reproducción de una cadena con su hermana sea igual que con cualquier otra, cosa que no existe en la naturaleza, salvo quizá por el tabú del incesto.

Si la repetición de un segmento en los dos progenitores no fuera señal suficiente de que se trata de un rasgo valioso, se podría hacer que cada entidad tuviese una dotación genética doble, (dominante y recesivo) y obligar a una coincidencia en los cuatro segmentos. En definitiva, se trata de tratar los segmentos *no-lineales* independientemente. Podríamos recorrer toda la población cada cierto tiempo, detectándolos. Una vez localizados, se puede provocar una microbúsqueda dentro del propio segmento *no-lineal* manteniendo constante la parte lineal, que por serlo, no afectará al peso resultante, hasta encontrar alguna combinación valiosa de los segmentos *no-lineales*.

El problema de la selección

En cada ciclo de un Algorimo Genético se selecciona un subconjunto de las soluciones o individuos existentes, eliminando el resto. Los individuos seleccionados se reproducirán entre sí, generando nuevas soluciones. La función que decide qué entidades serán seleccionadas, la llamada *función de evaluación* (FE), puede ser muy difícil o imposible de conseguir. En el **problema del viajante**, que debe recorrer varias ciudades hasta volver al punto de partida incurriendo en el mínimo coste, la FE calculará la suma de las distancias entre las ciudades recorridas en el orden especificado y devolverá la inversa de este valor.

Figura 3. Cada individuo posee una dotación genética doble.



En el **tres en raya**, hemos hecho que las entidades jueguen partidas entre ellas. La entidad ganadora recibirá un peso alto y la perdedora uno bajo, que podría variar en función de lo que ha tardado en perder. El caso de la **programación automática** es más complicado. ¿Cómo definimos qué es lo que debe hacer un programa, sin tener que escribir ese programa? Se podrían ofrecer varios pares de conjuntos de datos de entrada y salida, y suponer que el programa es correcto cuando genera los datos de salida esperados en todos los ejemplos. En general, los problemas en los que la meta buscada posea un caracter contínuo (en los que el objetivo se pueda cumplir en distintos grados) serán más fáciles de resolver que aquellos en los que la meta buscada es *todo-nada*, es decir, en los que o se consigue el objetivo completamente o no se consigue en absoluto. Finalmente, los problemas más espinosos de todos son aquellos en los que, al menos por ahora, la FE no es automatizable, como el caso de un programa capaz de crear música, ya que ¿dónde encontraremos a alguien dispuesto a escuchar y opinar acerca de interminables melodías inicialmente absurdas?

Conclusión

Los algoritmos genéticos tienen la ventaja de ofrecer un mecanismo adaptativo de resolución de problemas, de forma que, aunque el problema cambie, éste se pueda seguir resolviendo. "Resolver un problema cambiante", llevado al límite, es igual a "Resolver cualquier problema". Según este planteamiento podríamos pensar en diseñar un esquema común que facilite a los agentes aprender en cualquier entorno de problema. En el próximo artículo nos centraremos en el *aprendizaje* y veremos en profundidad los fundamentos de este esquema de aprendizaje válido para cualquier problema, basado en Vida Artificial y Algoritmos Genéticos.

DELPHI 3.0

Enrique de la Lastra

Delphi, la herramienta que ha revolucionado la programación, alcanza su madurez con la versión 3.0 introduciendo notables mejoras para el desarrollo de aplicaciones de bases de datos e Internet.

En los dos años de vida que tiene el compilador Delphi, no ha dejado de sorprendernos con las posibilidades que nos brinda. No sólo nos encontramos con una herramienta que facilitaba enormemente la programación Windows, sino que, además, se nos ofrecía un lenguaje sencillo y claro a la vez que potente, con mecanismos de programación orientada a objetos más simples que los proporcionados por C++ y un diseño orientado a componentes.

Ahora, con la nueva edición de Delphi, la 3.0, obtenemos un producto que después de pasar por la aprobación de los usuarios y de la industria, ha evolucionado y se ha asentado como una de las mejores opciones tanto para el programador amateur como para los equipos profesionales de aplicaciones Cliente/Servidor e Internet.

Y para los más escépticos sobre el futuro de esta herramienta, los datos hablan por sí solos: más de 600.000 copias vendidas avalan su valía y auguran un futuro prometedor. Compañías como la NASA, American Airlines, Arthur Andersen, BMW, AT&T, BT, BBC, y muchas más son clientes de esta aplicación.

En este artículo se verán las mejoras introducidas en el entorno de desarrollo (IDE: *Integrated Development Environment)* y los *Packages* (paquetes), librerías estilo DLL que reducen considerablemente el tamaño de los ejecutables. Se estudiarán las nuevas tecnologías de esta nueva edición de Delphi, para el desarrollo de aplicaciones cliente "pequeñas", lo que permitirá su fácil distribución, configuración y mantenimiento. Además de todo esto, conoceremos el soporte de las tecnologías respaldadas por Microsoft: COM y ActiveX (la respuesta al estándar CORBA y a los Java Beans).

Hay mucho más que ver y analizar, pero no es posible resumirlo todo en el espacio de un sólo artículo. Si es posible, lo analizaremos en un próximo artículo.

Características de Delphi 3.0

Delphi 3.0 incorpora una gran variedad de nuevas herramientas y mejoras, algunas de las cuales repasaremos en este artículo. Para esquematizarlas podemos agruparlas en cuatro grupos:

- Nueva interfaz e IDE: Veremos el nuevo aspecto y una nueva herramienta (*Code Insight,* una de las tecnologías *Insight,* que se comentan más adelante) que facilita la edición del código y mejora la productividad del programador

- Nueva arquitectura de la VCL: echaremos un vistazo a los fundamentos

Tabla 1. Características de las distintas versiones de Delphi 3.0 (Fuente: Borland).

CARACTERÍSTICA	E	P	C/S
Desarrollo visual RAD (Rapid Application Development) fácil	✓	✓	✓
Compilador de 32 bits (en código nativo y optimizado) más rápido del mundo	✓	✓	✓
Visual Component Library (VCL) con más de 100 componentes reutilizables, incluyendo controles de Windows 95	✓	✓	✓
IDE (Integrated Development Environment) profesional con Editor y Depurador integrados	✓	✓	✓
NUEVO: Expertos CodeInsight para agilizar la codificación y reducir errores de sintaxis	✓	✓	✓
Facilidad de creación de DLL's reutilizables y ejecutables independientes	✓	✓	✓
NUEVO: Compilador de paquetes (packages) que produce ejecutables muy pequeños (un ejecutable de 200K de la versión anterior, queda reducido a 9K).	✓	✓	✓
NUEVO: Interfaces que facilitan la compatibilidad con COM y ActiveX nativos	✓	✓	✓
Compatibilidad total con Windows 95/NT, Win32 API, ActiveX, tecnología Multi-hilo, OLE, COM, ISAPI (Internet Explorer), NSAPI (Netscape), CGI y HTML	✓	✓	✓
Creación y utilización de Controladores y Servidores de Automatización OLE	✓	✓	✓
Almacén de Objetos y Herencia de Fichas Visuales que permiten reutilizar código	✓	✓	✓
NUEVO: Creación inmediata de componentes visuales con las Plantillas de Componentes	✓	✓	✓
NUEVO: Compatibilidad con motores para múltiples bases de datos que facilita el acceso a cualquier sistema de base de datos	✓	✓	✓
NUEVO: Controladores nativos y rápidos para MS Access, FoxPro, Paradox y dBase	✓	✓	✓
NUEVO: Database Explorer mejorado para gestionar tablas, alias e índices visualmente	✓	✓	✓
Incluye la edición Delphi 1.0 para realizar desarrollos sobre Windows 3.1 (16 bits)	✓	✓	✓
NUEVO: Creación de controles ActiveX, compilados al 100%, de altas prestaciones, sin necesidad de módulos runtime		✓	✓
NUEVO: Con ActiveForm /ActiveDocument podrá adecuar en un solo paso sus aplicaciones para la Web		✓	✓
NUEVO: Informes integrados que mostrarán sus datos en presentaciones profesionales		✓	✓
NUEVO: Gráficos enlazados dinámicamente con los datos		✓	✓
Código fuente de la VCL con un manual impreso para desarrollar y personalizar componentes		✓	✓
Conectividad ODBC absoluta		✓	✓
Rejilla Multi-Objeto para aplicaciones avanzadas de base de datos		✓	✓
Diccionario de datos escalable para mantener la integridad de los datos		✓	✓
Actualizaciones con caché para mejorar la respuesta de los servidores		✓	✓
Internet Solutions Pack para adecuar aplicaciones para la Web		✓	✓
InstallShield® Express para crear instalaciones profesionales		✓	✓
Local InterBase® (licencia para un solo usuario) para desarrollo SQL sin conexión remota		✓	✓
Amplia documentación impresa		✓	✓
NUEVO: Aplicaciones multinivel de base de datos			✓
NUEVO: Aplicaciones cliente/servidor para la Web			✓
Juego completo de herramientas SQL			✓
Controladores nativos de altas prestaciones para Oracle, Sybase, DB2, InterBase, Informix y Microsoft SQL Server			✓

Leyenda: E: Estándar. P: Profesional. C/S: Cliente/Servidor

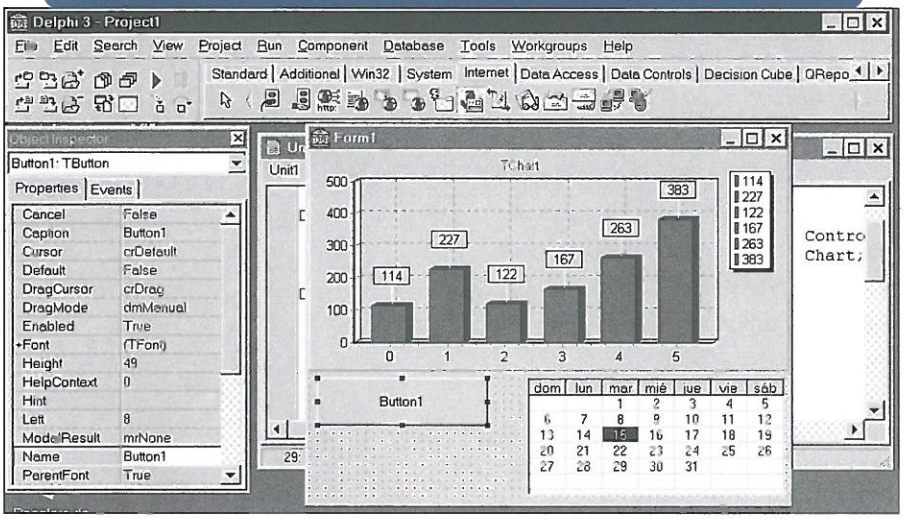
de la nueva estructura de la VCL y a algunos componentes nuevos. Se verá también la tecnología de "paquetes", que disminuyen drásticamente el tamaño de los ejecutables.

- Las tecnologías que Borland ha venido en llamar *"Insight"* (conocimiento profundo), aumentan la productividad en el desarrollo Windows.

Estas tecnologías son (además del *Code Insight)*:

1. *Business Insight:* Convierte los datos en algo "vivo", fácil de analizar.

Figura 1. Nueva interfaz de Delphi 3.0.



2. *Active Insight:* conjunto de expertos y herramientas para la creación de componentes *ActiveX* y derivados.

- Las tecnologías *"Multi-Tier"* (multinivel, es decir de varios escalones o niveles), con las que se reduce el coste de configuración y distribución mediante aplicaciones clientes pequeñas (o delgadas), en entornos de trabajo con tiempos críticos:

3. *Remote Data Broker* (Agente de datos remoto): permite aplicaciones *Multi-Tier* de configuración nula.

4. *Business Broker* (Agente de la aplicación): otorga seguridad ante fallos a entornos de aplicaciones críticas.

5. *Constraint Broker* (Agente de restricciones): promueve la integridad de los datos y con ello la reducción del tráfico en la red.

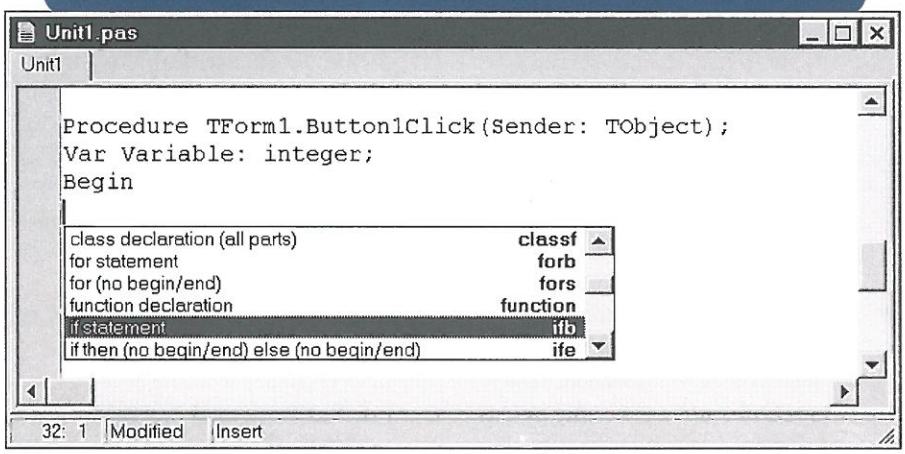
Otro conjunto de tecnologías importante, que sin embargo no explicaremos en este artículo (debido a que disponemos de un espacio limitado), son las tecnologías que se utilizan para suministrar información a través de la WWW (*World Wide Web,* Red de enlaces extendida por todo el mundo), partiendo de aplicaciones diseñadas para entornos Cliente/Servidor considerados tradicionales (como el *Web Broker* y el *Web Deployment).*

En la tabla 1 se presenta un resumen de las características de Delphi 3.0 en cada una de las tres versiones en las que se vende, indicando cuáles son nuevas respecto a la anterior edición de Delphi.

En cuanto a los requisitos mínimos que requiere la aplicación, son los siguientes:

- Procesador 486DX/66 MHz (se recomienda Pentium o similar).
- Windows 95 o Windows NT.
- Memoria: 12 Mb de RAM (se recomienda al menos 16 Mb).
- 60 Mb de espacio libre en el disco duro.
- Unidad CD-ROM.

Figura 2. Plantillas de sentencias de código.



Nuevo IDE

La interfaz de Delphi 3.0 es prácticamente idéntica a la de Delphi 2.0, pero, siguiendo la línea iniciada con el Internet Explorer, los botones de las barras de herramientas son ahora "planos" (sólo toman forma tridimensional al pasar el ratón por encima de ellos). Podemos observar esta nueva interfaz en la figura 1.

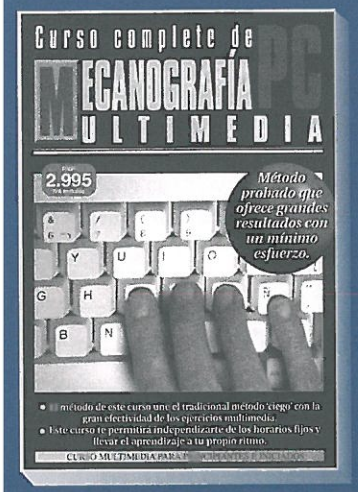
Otra novedad es la aparición de un panel vertical a la izquierda de cada *unit* abierta, que permite fijar fácilmente los *breakpoints* (puntos de ruptura) con sólo hacer "click" en dicho panel. Además muestra las líneas para las que se ha generado código máquina.

Pero la mayor novedad en el IDE es el *Code Insight. Code Insight* constituye un conjunto de herramientas (*wizards*, o magos, lo que en la antigua terminología eran "asistentes") que han sido desarrolladas con el propósito de facilitar la edición del código y consiguen reducir el número de consultas a la ayuda *on-line.* Pasamos a estudiarlo en el siguiente punto.

Curso completo de MECANOGRAFÍA PC MULTIMEDIA

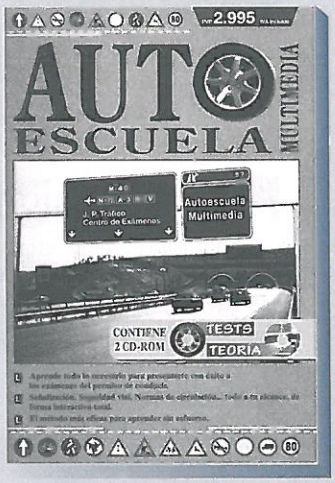
P.V.P
2.995
Ptas. c.u.
IVA Incluido

- El método de este curso une el tradicional método 'ciego' con la gran efectividad de los ejercicios multimedia.
- Este curso te permitirá independizarte de los horarios fijos y llevar el aprendizaje a tu propio ritmo.



AUTO ESCUELA MULTIMEDIA

- Aprende todo lo necesario para presentarte con éxito a los exámenes del permiso de conducir.
- Señalización, Seguridad vial, Normas de circulación... todo a tu alcance, de forma interactiva total.
- El método más eficaz para aprender sin esfuerzo.



ABETO EDITORIAL

Envíe este cupón por correo o fax (91) 661 43 86 o llamando al teléfono: (91) 661 42 11* de lunes a jueves de 9:00 a 14:00 y de 15:00 a 18:00 h., y viernes de 9:00 a 15:00 h.

Deseo que me envíen: ☐ AUTOESCUELA MULTIMEDIA **por 2.995 ptas. + 250 de gastos de envío**
☐ MECANOGRAFÍA PC MULTIMEDIA **por 2.995 ptas. + 250 de gastos de envío**

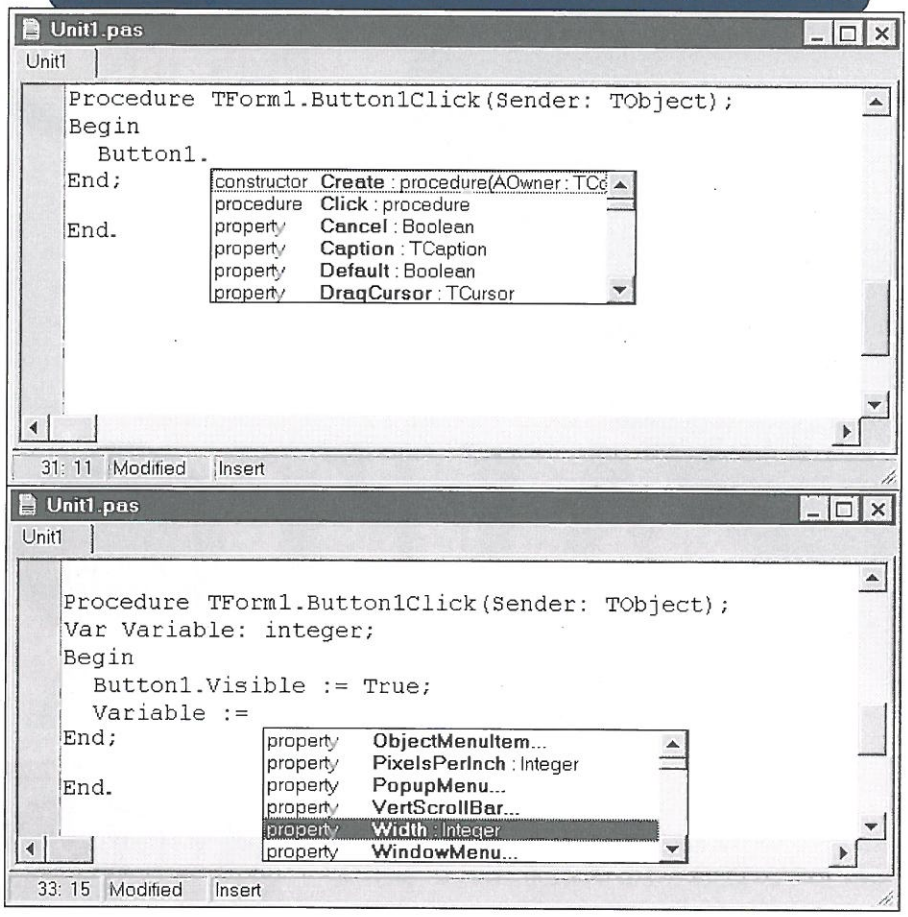
Nombre y apellidos ..... Domilicio .....
Población ..... C.P. ..... Provincia .....
Teléfono ..... Edad ..... Profesión .....

FORMA DE PAGO:

☐ Talón a ABETO EDITORIAL ☐ Giro Postal (adjunto fotocopia de resguardo)
☐ Contra-reembolso ☐ VISA nº _____ _____ _____ _____ Cad. __ __

ABETO EDITORIAL
C/ Aragoneses, 7
28108 Alcobendas (Madrid)

Figura 3. Wizard que facilita la edición de métodos o propiedades aplicables a cada objeto. También muestra las variables o métodos que se pueden asignar a una variable.



Code insight

Es un conjunto de cinco herramientas que nos ahorran mucho tiempo, tanto de edición de código como de consulta:

1. *Code Template Wizard* (Mago para la generación de sentencias de código). Consta de una serie de plantillas que permiten la introducción de un conjunto de líneas de código con sólo escribir unos pocos caracteres o seleccionarlas de una lista que aparece pulsando **Ctrl+J**.

   Por ejemplo (véase la figura 2), tecleando "ifb" aparecerá automáticamente una sentencia "if". Otra opción será pulsar Ctrl+J, aparecerá entonces el cuadro mostrado en la figura 2, y bastará hacer clic en la línea "if statement**ifb**". Todos las plantillas son configurables y se pueden añadir cuantas plantillas personalizadas queramos.

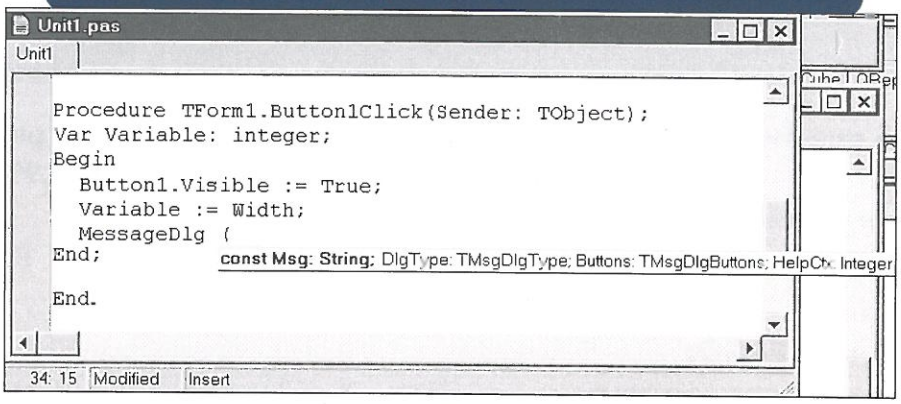
1. *Code Completion Wizard* (Mago para la edición de las propiedades). Al introducir un nombre de un objeto en el editor, seguido de un punto, aparece una ventana superpuesta con todos los métodos y propiedades de dicho objeto. Sólo hay que hacer clic en el que queramos (figura 3). De igual manera, después de escribir el nombre de una variable seguido de ":=" aparecerá una ventana con todas las posibles variables y objetos que se pueden asignar. Se puede forzar la aparición de estas ventanas pulsando la combinación: **Ctrl +Espacio.**

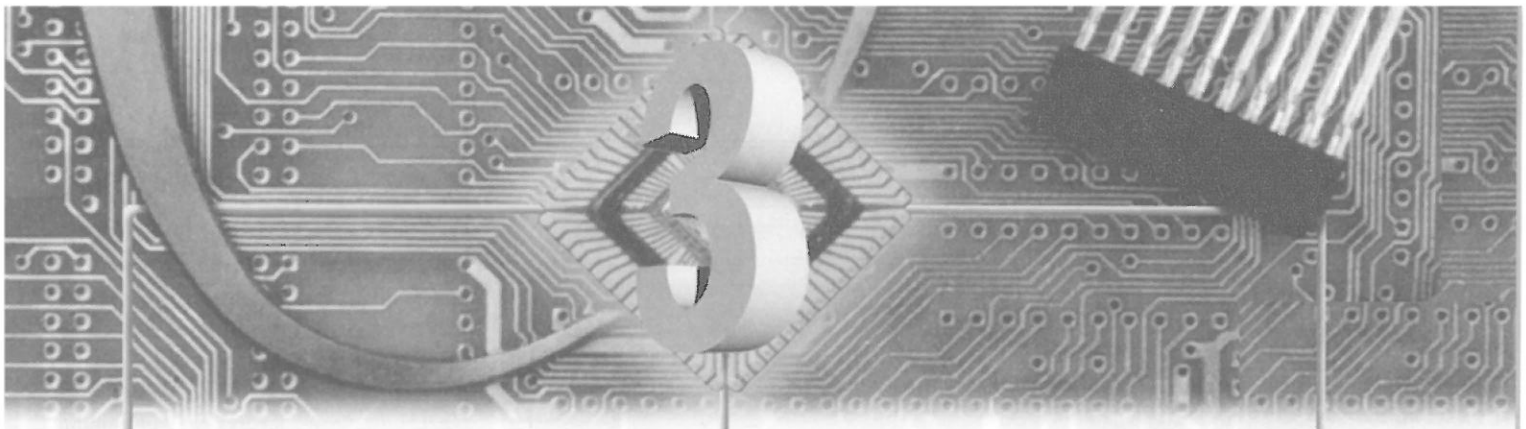
1. *Code Parameter Wizard* (Mago para la edición de parámetros de métodos y procedimientos). Al teclear un nombre de un método o un procedimiento, seguido de un paréntesis "(", se muestra una ventana con los parámetros de ese procedimiento y sus tipos.

Se puede forzar la aparición de estas ventanas pulsando la combinación: Ctrl +Espacio

El primer parámetro que debe ser introducido está en negrita, y según se van añadiendo los parámetros se va poniendo en negrita el parámetro siguiente (figura 4). De la misma

Figura 4. Wizard para simplificar la edición de los parámetros de métodos o procedimientos.





PROYECTO COLON

www.colon.net]

[www.seric.es]

www.QRZ.net

[www.qrz.net]

Disponemos de tres Servidores de INTERNET

Seric, empresa consolidada desde 1987, con amplia experiencia en el sector de las comunicaciones le ofrece un amplio abanico de **soluciones** para la empresa y el usuario particular ajustándonos a sus necesidades

TARIFAS COLON

- Conexión a INTERNET hasta el 31 de Enero de 1998 (6 meses)
- Cuenta de correo electrónico
- Curso interactivo en CD-Rom

4.000 ptas.

I.V.A. incluido

TARIFAS

Línea R.T.B. 25.000 ptas.+I.V.A.
- Cuenta de correo
- 1 Mb para sus páginas Web

Línea R.D.S.I. 75.000 ptas.+I.V.A.
- Cuenta de correo
- 2 Mb para sus páginas Web

TARIFAS www.QRZ.net

Línea R.T.B. 15.000 ptas.+I.V.A.
- Cuenta de correo
- 2 Mb para sus páginas Web

Línea R.D.S.I. 35.000 ptas.+I.V.A.
- Cuenta de correo
- 4 Mb para sus páginas Web

85.000 ptas.+I.V.A.
- Alta y creación de dominio
- 4 direcciones E-mail
- Espacio ilimitado en disco

Telefónica Consultor homologado

Seric le tramitará el cambio de RTB a RDSI, o le gestionará el alta de una nueva línea RDSI, consúltanos

SeriC's Room

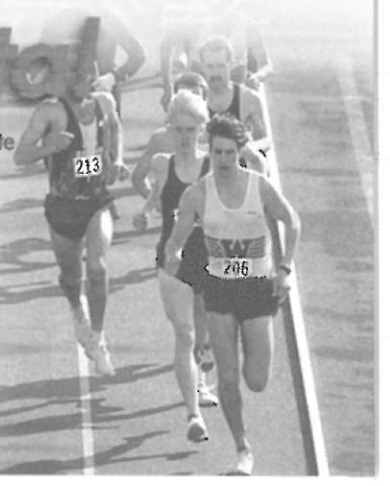
A ROM WITH A VIEW

Passeig Prim, 27 baixos Tel.977/ 33 11 40 Fax.977/ 33 11 14 – 43202 Reus http://www.seric.es - E-Mail: ventas@mail.seric.es

stollmann ¡Velocidad Punta!

Para más información
http://www.stollmann.com

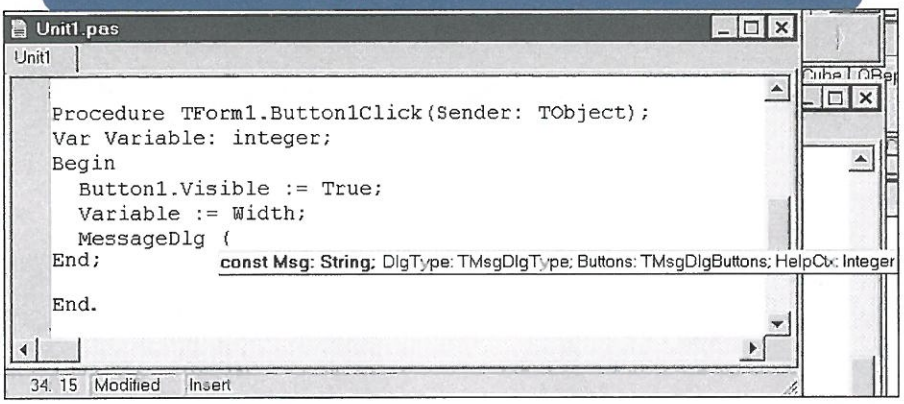
- Adaptador RDSI de alto rendimiento, fácil instalación, bajo coste y totalmente externo sin necesidad de manipular el equipo.
- Compatible con MS-DOS, Windows 3.X/95 NT, UNIX, MAC.
- TA+PPP Contiene todo lo que un adaptador RDSI necesita tener.



Datos Trasmisión 115.000 bps.

32.000 ptas. + I.V.A. Alimentador, manual, cable conexión serie y cable conexión línea RDSI.

Figura 5. Paquetes (de tiempo de diseño) instalados y opciones de compilación de paquetes en el ejecutable.



manera, se puede forzar la aparición de esta ventana mediante la combinación de las siguientes teclas: **Ctrl+Shift+Espacio**.

1. *Tool Tip Expresion Evaluation* (Evaluador de valores de variables, propiedades, constantes o parámetros). Ésta es una de las mejoras más importantes introducidas en el depurador integrado de Delphi, que fortalecen una de las partes más flojas de las anteriores ediciones. Cuando se está procediendo a la ejecución de un programa, no hace falta más que pasar el ratón por encima de una propiedad para que aparezca una ventanita (estilo *Hint*) con su valor actual.

Aunque incorporáramos pocos controles en una aplicación, el tamaño del ejecutable era como mínimo de unos 200Kb

2. *DLL Debugging* (Depuración de librerías DLL dentro de Delphi). Esta herramienta consigue solventar otra de las deficiencias que tenía Delphi. Ahora ya es posible depurar tanto ejecutables como DLL's (¡por fin!) sin tener que pagar por un producto adicional (como Turbo Debugger).

Nueva VCL. Packages

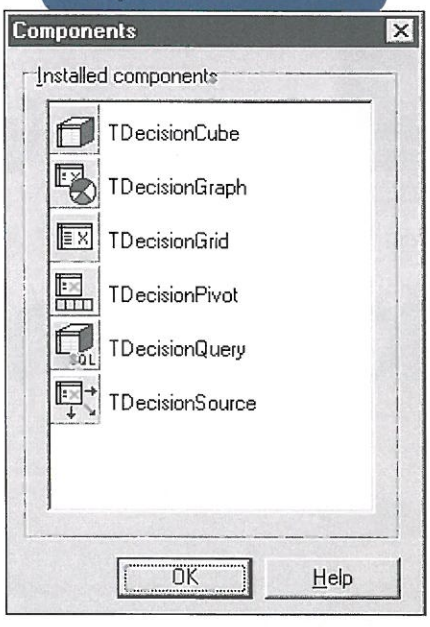
La VCL de Delphi 3.0 se ha ampliado con gran variedad de componentes entre los que se encuentran varios desarrollados por terceras personas que han sido adquiridos por Borland (entre los que se encuentra el *Quick Report* y el *Tee Chart*, que se comentan más adelante).

Se incluye una novedad: los *Packages* (paquetes). Uno de los "problemas" de Delphi era que aunque incorporásemos muy pocos controles a una aplicación, el tamaño del ejecutable era como mínimo de unos 200Kb (debido a que se compilaba en el ejecutable buena parte de la librería de componentes). Aunque hoy día esto no supone un problema grave (ni siquiera pequeño, ateniéndonos a los precios actuales de los discos duros y las memorias) el hecho de duplicar el contenido de la VCL en todos los ejecutables no era lo más indicado. Los paquetes se han creado para solventar este inconveniente y no son sino librerías DLL formadas por la compilación de varias *units* de VCL's. Cada paquete puede contener tantas *units* como queramos, de forma que podemos distribuir las *units* de la VCL dentro de más o menos paquetes, según nos parezca conveniente. La extensión del archivo de un paquete no es *.DLL sino *.DPL (*Delphi Package Library*), ya que aunque tiene la estructura de una DLL, las referencias a los métodos y propiedades se realizan normalmente, sin necesidad de hacer ningún vínculo ni llamada especial al archivo *.DPL.

Las ventajas de los paquetes se acentúan cuando el número de aplicaciones Delphi instaladas en un ordenador aumenta (ya que el código de los paquetes es compartido). Además cuando se cambia un componente o se instala uno nuevo, ya no hace falta recompilar toda la librería de componentes (de hecho la librería de componentes como tal ha sido sustituida por los paquetes) sino que basta recompilar el paquete que contenga la *unit* de dicho componente, lo cual acelera mucho el proceso. Existe una limitación referente a que un ejecutable no puede incluir paquetes que contengan *units* repetidas.

Seleccionando la opción "Options..." del menú "Project" aparece el cuadro de

Figura 6. Componentes incluidos en el paquete DCLDSS30.DPL (Paquete del Decision Cube).



la figura 5, donde se pueden observar los paquetes de tiempo de diseño instalados (que son los paquetes instalables en Delphi, ya que los de tiempo de ejecución son la materia prima de la que está implementado Delphi 3.0). También se pueden modificar desde este diálogo los componentes que incluye cada paquete (para ello hay que pulsar el botón: "Components", ver figura 5). Por ejemplo, en la figura 6 se presentan los componentes incluidos en el paquete del *Decision Cube.*

Para instalar o desinstalar un paquete basta hacer "click" en la casilla de marca correspondiente (figura 5), y tras una pequeña recompilación se añadirá o eliminará la pestaña correspondiente de la paleta de componentes.

Si queremos que nuestro ejecutable compile como en anteriores versiones (es decir, añadiendo el código de los componentes al ejecutable), basta con dejar sin marcar la casilla de marca que tiene la etiqueta: "Build with runtime packages" (actuando así, obtendremos un ejecutable mucho más abultado). Si por el contrario queremos sumarnos a los beneficios ofrecidos por esta nueva técnica, marcaremos dicha casilla.

Algunos componentes nuevos son:

1. *TSpeedButton:* no es un componente nuevo, pero incorpora una propiedad que lo convierte en novedad. Esta propiedad es *flat* (plano), que permite mostrarlo con la nueva presentación de los botones de la interfaz de Delphi 3.0.

2. *TDataSet:* tampoco es nuevo, pero ésta versión elimina las dependencias con el BDE (*Borland Database Engine*), que ahora se añaden a una clase hija de ésta: *TBDEDataSet* (o *TDBDataSet*, que hereda de *TBDEDataSet*)

3. *TSplitter:* es un panel que presenta una línea divisoria horizontal o vertical, y permite cambiar el tamaño de las partes desplazando esa línea.

4. *TAnimate:* después de los múltiples componentes *freeware* y *shareware* que aparecieron en versiones anteriores para realizar animaciones, ahora se nos regala este componente de animación que encapsula el control Win32 ANIMATE. Soporta todas las típicas animaciones predefinidas de Windows 95 así como cualquier fichero AVI sin sonido.

5. *TToolBar*: barras de herramientas del estilo de las del propio Delphi 3.0 o del Microsoft Internet Explorer.

6. *TCheckListBox:* otro componente ya existente como *shareware*, que ahora se añade a la paleta estándar. Es un ListBox igual que los habituales pero que permite marcar cada elemento de la lista.

7. *TStaticText:* igual que un TLabel pero con un *handle.*

8. *TDateTimePicker:* se acabaron los analizadores sintácticos que nos teníamos que currar para comprobar que las fechas o las horas se introducían correctamente. Este componente nos resuelve el problema (aunque tiene el inconveniente de que necesita la DLL: COMCTL32.DLL de la versión 3.0 de Internet Explorer).

Business Insight

Este término engloba la nueva arquitectura de componentes para bases de datos y ayuda a las decisiones que, trabajando de manera conjunta y uniforme, facilitan los procesos de análisis de datos corporativos o aquéllos provenientes de Internet.

Está pensado para las aplicaciones de empresa en las que se genera y trata gran cantidad de información, que siempre es difícil transformar en información útil.

Para ello, se incluyen tres componentes muy integrados entre sí que permiten mostrar gráficamente los datos:

1. *Decision Cube:* análisis multidimensional de datos.

2. *TeeChart:* presentación gráfica de la información.

3. *Quick Reports:* informes rápidos sobre los datos.

Decision Cube

Conjunto de componentes que ayudan en la generación de gráficos y diagramas de barras, facilitando el análisis de datos al suministrar distintas vistas y resúmenes de los datos así como mecanismos de *drag and drop* (arrastrar y soltar) e integración con gráficos.

El conjunto de componentes de *Decision Cube* se puede observar en la figura 6. El componente (no visual) TDecisionCube es un almacén multidimensional de datos que está conectado a un *dataset* (generalmente una expresión SQL en un TQuery, o un TDecisionQuery). El TDecisionQuery, añade un editor para configurarlo fácilmente. Existe un componente TDecisionSource conectado al TDecisionCube que representa el estado actual.

Tee Chart

Gráficas, diagramas de barras, de líneas o en modo "empanada", representaciones polares, etc. (hasta once tipos distintos) tanto en dos dimensiones como en 3-D. Una serie de componentes magníficos que fueron desarrollados por un programador español (David Berneda) y que han sido adquiridos por Borland e incorporados a la paleta de componentes estándar de Delphi 3.0. Permiten tanto la visualización de datos generados por programa (es

decir mediante código) como datos importados de una base de datos.

Además, estos gráficos son dinámicos, es decir, cambian al ritmo que lo hacen los datos (si se actualiza un registro en la base de datos, automáticamente se produce la correspondiente modificación en la gráfica asociada). Se integran perfectamente con el diseñador de informes *Quick Report* (ver siguiente punto), ya que incorpora un componente específico que enlaza con el de *Quick Report*. Son realmente sencillos de usar (gracias a una interfaz de diseño muy elaborada y atractiva visualmente) y de aprender.

Quick Report es un conjunto de componentes diseñados para el desarrollo rápido de informes complejos

Estos componentes combinan la velocidad propia de los controles de Delphi con la flexibilidad del desarrollo rápido de aplicaciones Cliente/Servidor e Internet. Facilitan la labor de análisis de grandes volúmenes de datos (tanto estáticos como dinámicos) y, por tanto, la toma de decisiones en base a éstos.

Quick Report

Conjunto de componentes diseñados para el desarrollo rápido de informes complejos tanto para su publicación en la Web como para su distribución con aplicaciones "normales". Las características de estos componentes incluyen: informes en columnas y maestro-detalle, más de 130 informes de etiquetas para *mailings*, impresión, previsualización en la propia aplicación de lo que va a aparecer en la hoja impresa, definición de la orientación de la hoja, exportación a formatos texto y HTML (para distribución de los datos en Internet).

Como se comentó en el apartado anterior, se integran con *Tee Chart* y al tener arquitectura *multi-thread* (multihilo de ejecución) ofrecen un gran rendimiento.

La interfaz es sencilla e intuitiva, estando apoyada con varios *wizards* (magos).

Active Insight

En Delphi 3.0 encontramos todas las herramientas necesarias para la creación de componentes "ActiveX" (que se ha convertido casi en un estándar de la industria informática) y objetos "COM", lo que permite una total compatibilidad con lenguajes como C++, Java, Java Script, Power Builder o Visual Basic.

La creación de componentes ActiveX es tan sencilla con Delphi, que basta escoger cualquier componente estándar de la paleta y convertirlo en un ActiveX con las herramientas del compilador (¡sin necesidad de codificar una sola línea de código!). Sólo con esto ya tenemos a nuestra disposición una gran cantidad de posibles componentes ActiveX. Teniendo en cuenta además, que la mayoría de los desarrolladores de Delphi crean sus propios componentes, se tendrá siempre la posibilidad de convertirlos en ActiveX y así, tenerlos listos para compartirlos con otros equipos de desarrollo software (aunque éstos estén utilizando otra herramienta de programación) o para utilizarlos en cualquier navegador (Netscape Navigator o Microsoft Internet Explorer).

Para conseguir lo explicado anteriormente, Delphi trae consigo una serie de "Expertos" que se encargan de todo el trabajo. A diferencia de Visual Basic, no sólo se pueden crear "Formularios Activos" (es decir, formularios *ActiveX* contenedores de otros controles, como botones, casillas de marca, etc.) sino también, controles *ActiveX* totalmente orientados a objetos y generados en código máquina, esto es, independientes de cualquier módulo *runtime*. Con esto se consiguen objetos ActiveX de los que podemos heredar y añadir propiedades o métodos (lo cual es imposible en Visual Basic por no tener una programación orientada a objetos auténtica).

Soporte de objetos COM y DCOM

La tecnología COM está implementada en Delphi como un tipo nativo para permitir la máxima interoperabilidad y reutilización, consiguiendo, de este modo, soporte para otros estándares de Microsoft: *ActiveX, ActiveForm, ActiveDocument* y *ActiveContainer*. La tecnología COM es la respuesta de Microsoft a la especificación CORBA, mientras que los *ActiveX*, basados en COM, compiten con los objetos *Java Beans,* basados en CORBA.

La interfaz de los objetos COM tiene asociada un contador que lleva la cuenta del número de instancias activas

Las clases COM de Delphi 3.0 implementan las interfaces estándar OLE y COM y facilitan al programador la creación de aplicaciones Cliente/Servidor tales como servidores OLE de documentos y controles ActiveX, sin sufrir

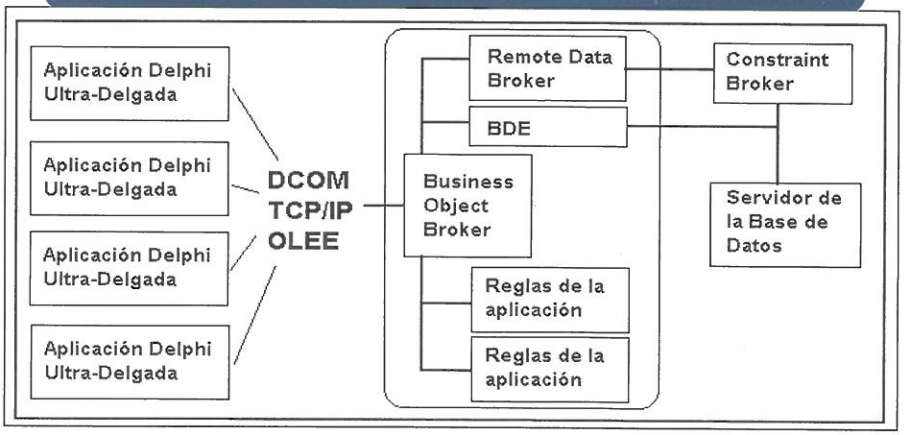
las limitaciones que vienen impuestas por los módulos *runtime*.

La interfaz de los objetos COM tiene asociada un contador que lleva la cuenta del número de instancias activas. Cuando este número alcanza el cero, la implementación de la interfaz de dicho objeto es automáticamente destruida (y la memoria liberada).

Arquitectura MULTI-TIER

Esta arquitectura está basada en la separación de la interfaz de usuario, la lógica de la aplicación y la lógica de acceso a los datos. Para entenderla, mejor comenzamos explicando la arquitectura *One-Tier*.

Figura 7



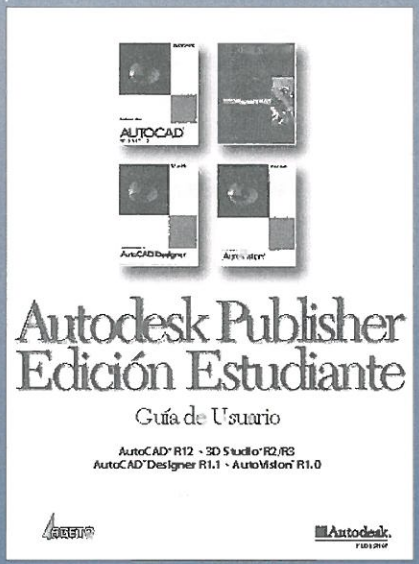
En las aplicaciones habituales, implementadas para máquinas aisladas, la interfaz de usuario (es decir lo que el usuario "ve" de la aplicación), el acceso a los datos (reglas y lógica de acceso a los datos) y la aplicación en sí (lo que se ejecuta por debajo de la interfaz de usuario), forman un todo (un ejecutable). Trabajando así, la programación es más sencilla y además confiere un control completo sobre la aplicación, pero tiene el inconveniente de que para modificar cualquiera de las tres partes lógicas, hay que recompilar todo el ejecutable y probablemente

AUTODESK PUBLISHER EDICIÓN ESTUDIANTE

¿QUÉ PROGRAMAS INCLUYE?

- AutoCAD R12
- 3D Studio R2/R3
- AutoCAD Designer R1.1
- Autovision R1.0.

Y además:
Más de 700 páginas de lecciones de Autodesk con guías de referencia y documentación software on-line



Formato 21x29,7

TODO LO QUE NECESITA PARA EL DISEÑO Y VISUALIZACIÓN PROFESIONAL EN 2D Y 3D LO HALLARÁ EN ESTE PAQUETE DE PROGRAMAS.

EDICIÓN ESPECIAL PARA ESTUDIANTES
19.500 PTAS. (I.V.A. INCLUIDO)

DE VENTA EN LIBRERÍAS Y DISTRIBUIDORES AUTODESK

modificar alguna de las otras partes (debido fundamentalmente a que no son totalmente independientes).

Este problema dio como idea la separación de la aplicación que accede a los datos, de la lógica y los permisos de acceso a los mismos. Así nació la, ya tan célebre, arquitectura Cliente/Servidor, que consiste en la separación de la aplicación en dos partes: una, el servidor de los datos y otra (otras) el cliente (los clientes) de esos datos. El cliente será una aplicación que solicita a la aplicación servidora los datos almacenados en una base de datos compartida. En esta nueva forma de trabajo, el cliente sigue teniendo la mayor parte del grueso de la aplicación: interfaz de usuario, aplicación y lógica de acceso a los datos. El servidor se encarga tan solo de administrar y actualizar los datos.

El siguiente paso es dividir las tres partes lógicas antes mencionadas e implementarlas por separado: en esto consiste la tecnología *Multi-Tier* (que en realidad es *Three-Tier,* tres niveles). Ahora, al estar separadas, estas tres partes pueden estar distribuidas en distintos ordenadores. En esta arquitectura, la aplicación cliente es muy pequeña (*thin client*), minimizándose el flujo de datos entre el cliente y el servidor, para cargar al mínimo la red.

Como se dijo en la introducción, el desarrollo *Multi-Tier* se apoya en Delphi 3.0 en los siguientes componentes: *Remote Data Broker* (Agente de datos remoto), *Business Broker* (Agente de la aplicación) y *Constraint Broker* (Agente de restricciones).

Remote Data Broker

Permite a los clientes comunicarse en un entorno *Multi-Tier,* reduciendo el tráfico en la red pero manteniendo la arquitectura Cliente/Servidor. Los clientes son ahora aplicaciones pequeñas (ya que no es necesario que incluyan el BDE), que se comunican con el BDE que está localizado en un servidor central. De esta manera el BDE sólo se configura en el servidor (figura 7).

Los datos recibidos por el cliente se pueden almacenar en la memoria o en el disco duro, reduciendo el número de conexiones al servidor y por tanto disminuyendo la carga de la red. La aplicación cliente puede actuar sobre sus datos locales con las herramientas habituales de acceso a BDs de Delphi.

El BDE detecta los posibles errores de transmisión, datos inválidos o fallos en la BD

Cuando el cliente realiza una modificación sobre los datos, sólo las modificaciones se envían de vuelta al servidor (y no todos los registros), reduciendo nuevamente el tráfico ofrecido a la red. El BDE, residente en el servidor, es el encargado de detectar los posibles errores de transmisión, datos inválidos o fallos en la propia base de datos. Una vez encontrado el problema, se informa a la aplicación del servidor, que a su vez enviará los mensajes de error que correspondan a los clientes.

Business Broker

El *Remote Data Broker* utiliza la metodología de orientación a objetos para crear objetos COM o servicios RPC (*Remote Procedure* Call). *Business Object Broker* suministra la localización transparente de cualquiera de estos objetos (servidores RPC, objetos COM, reglas de la aplicación o Agentes remotos). En lugar de unir el cliente con el objeto distribuido remoto, el cliente enlaza con un servicio que es el que localiza el objeto requerido y comprueba la disponibilidad de dicho objeto en todas las máquinas.

Constraint Broker

Es el agente que centraliza en el servidor las reglas de integridad de los datos, evitando que dichas reglas residan en todos y cada uno de los clientes (lo que "engordaría" las aplicaciones cliente con mecanismos duplicados en todas ellas).

Conclusiones

Hemos repasado en este artículo algunas de las novedades de Delphi 3.0. Si el producto era bueno, en esta nueva edición es francamente superior, no sólo por su avanzada tecnología, sino sobre todo por la cantidad de nuevas herramientas que incorpora. Delphi ya no es una promesa que gustó a muchos programadores; es una realidad, un compilador completo, tanto para el entorno particular como para el profesional. Sin embargo, nos hemos dejado muchas cosas en el tintero: el repositorio de objetos, las herramientas Web e Internet, las facilidades para el desarrollo Internet a partir de aplicaciones Cliente/Servidor, el nuevo BDE, las aplicaciones de consultas SQL, ...

Contactar con el autor

Si alguien desea enviar algún comentario sobre este o sobre los anteriores artículos, puede hacerlo enviando un e-mail a la dirección de la revista: solop@towercom.es o a la dirección del autor: miluen@redestb.es indicando como "subject" del e-mail: "DELPHI SOLOP" (en mayúsculas y con un espacio entre ambas palabras).

Especial Linux, programas de ajedrez, Site Visual Basic y mucho más

CONTENIDO DEL CD

En este CD-ROM que os presentamos podéis encontrar aplicaciones Linux variadas, la última entrega del especial sobre Delphi, Site Visual Basic de FarPoint, varios programas de ajedrez, la gama antivirus de McAfee y mucho más.

Linux

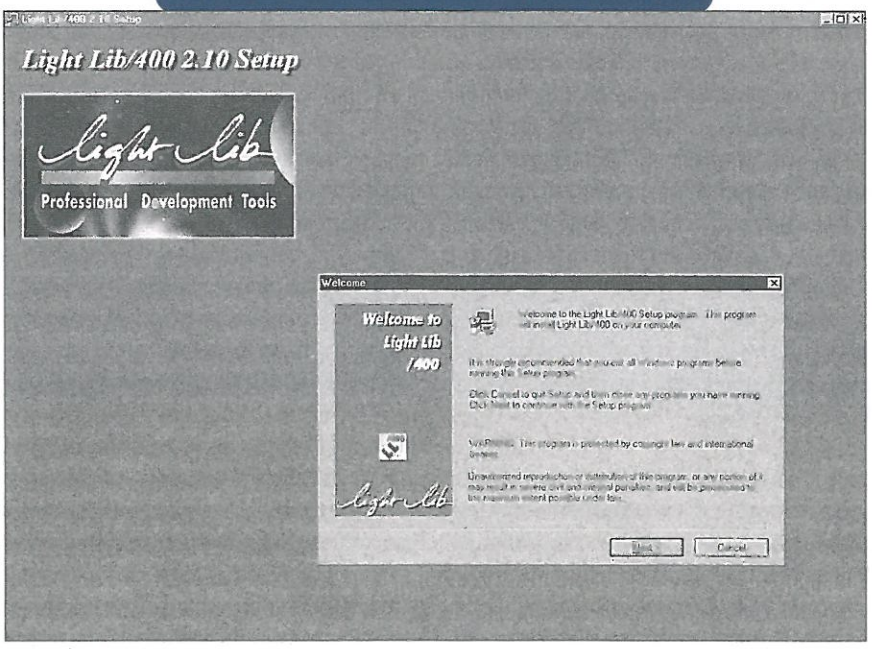
Teniendo en cuenta que este número está dedicado especialmente a la versión más popular de Unix para PCs, hemos querido incluir las aplicaciones más representativas de cada campo pero programadas para ejecutarse en Linux. Como el lector podrá apreciar algunas de ellas no tienen nada que envidiar a sus hermanas mayores escritas para entornos comerciales como el Motif e incluso ganan en versatilidad a verdaderas suites escritas para entornos Windows.

Entre ellas podríamos destacar al Acrobat Reader, famoso visualizador de ficheros .pdf especiales para editar documentaciones de programas ahorrando en papel; el PovRay, un programa para Ray-Tracing con el que podremos crear escenas tan realistas como queramos; o el SciLab, con el que tendremos a nuestro alcance la física y las matemáticas. También hemos incluido las librerías gráficas Xforms; una hoja de cálculo que no tiene nada que envidiar al Excel, WingZ; un visualizador que soporta cualquier formato de imágenes, el ImageMagick; así como un editor especial para programadores como es el Xcoral; o un editor de páginas HTML, asWedit. Por supuesto seguiremos incluyendo aplicaciones Linux para aquellos que todavía no están enamorados del verdadero sistema operativo de libre distribución, el Linux.

Delphi

Nuestro especial de Delphi no estaba completo sin las aplicaciones que incluimos este mes en la revista, que demuestran la poderosa incursión de Delphi en el mundo comercial y de los ordenadores de mediano tamaño.

FIGURA 1

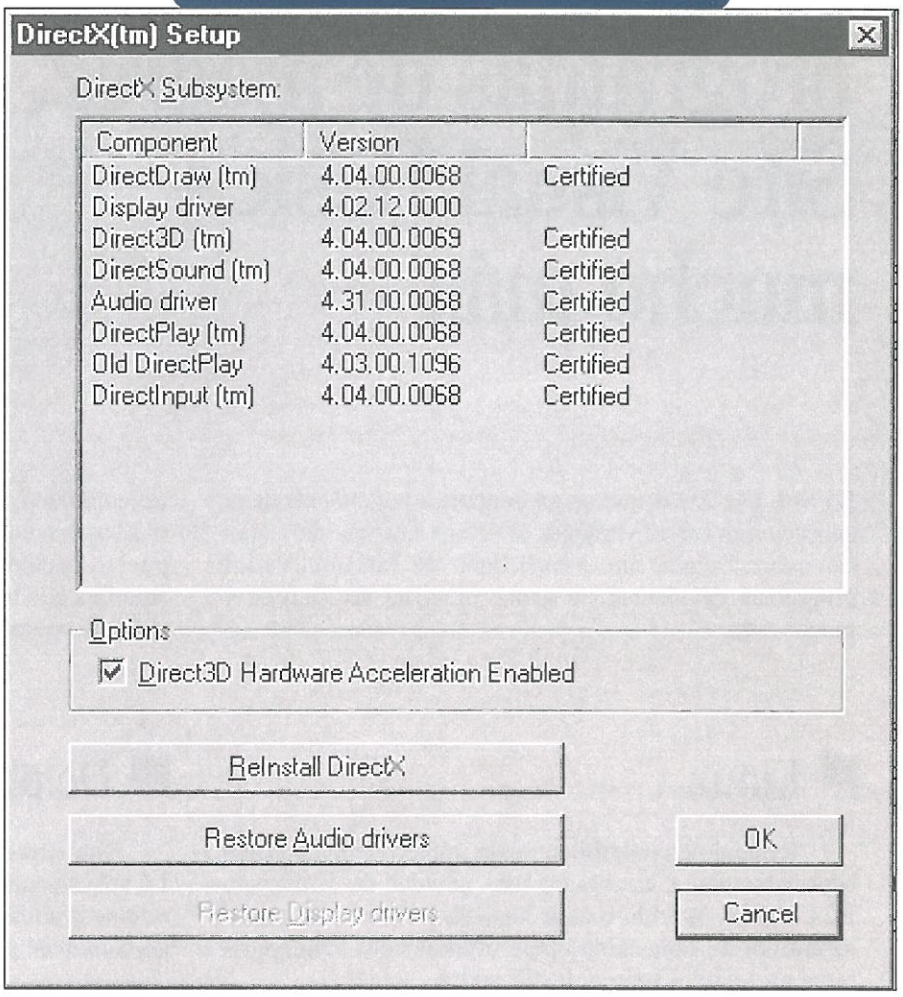


El primer paquete que incluimos, el LightLib/400, es una herramienta en Delphi para acceder de manera transparente a entornos AS/400 de IBM, tanto a sus recursos, como bases de datos nativas de esta máquina como a sus programas vía Remote Procedure Control. Existe, además, una versión en VCL y otra en OCX para facilitarnos el trabajo a la hora de crear directamente nuestras aplicaciones.

También incluimos una potente herramienta de creación visual de informes, QuickReport, y algunas aplicaciones y controles escritos en Delphi, que demuestran su uso como BarComp, un manejador de códigos de barras.

Dejándonos llevar por la era visual, Orpheus nos proporciona una moderna biblioteca de estos componentes visuales para Delphi que compilan en Borland Delphi de 16 y 32 bits.

FIGURA 2



Visual Basic

Hemos visitado la sede Web de FarPoint, un interesante desarrollador de componentes, controles y aplicaciones escritas en Visual Basic, y otros. Ponemos a tu disposición las versiones de evaluación de los productos más punteros de esta marca, con los que hacer aplicaciones de gran calidad técnica y aspecto profesional.

Redes

Dos son las aplicaciones que hemos incluido en este número, una es el NetShield y otra es el Netatalk, la primera de la marca McAfee, está desarrollada para la detección de virus en una red de ·rea local. La otra es un entorno de emulación de la red de Apple AppleTalk pero que se ejecuta sobre Linux.

En próximos números incluiremos el WebScan de McAfee, especial para incursiones virulentas en nuestro servidor Web.

Varios

DirectX en su versión 3.0a es lo último en aceleración de gráficos para Windows de 32 bits, estas librerías le sacan el máximo partido a nuestra tarjeta gráfica. En un número próximo pensamos presentaros el entorno de desarrollo de Microsoft para estas librerías, imprescindible para el auténtico programador de juegos.

Además nos hemos dado una vuelta por el Site de McAfee, y nos hemos traído lo último en virus. En subsiguientes entregas os ofreceremos las versiones shareware de los programas que desarrolla esta marca pero que no tienen que ver con virus, veréis que no sólo de virus vive McAfee.

Índice y Fuentes

Para completar la entrega, disponéis del índice actualizado así como de las fuentes que se mencionan en los artículos

Hay otra forma de ver el futuro.

SIMO TCI

FERIA INTERNACIONAL DE INFORMATICA, MULTIMEDIA Y COMUNICACIONES.

Ven a ver las nuevas tendencias que marcarán el próximo milenio.
Ven a SIMO TCI 97. Verás futuro.

MADRID 4-9 NOVIEMBRE 1997

Incluye:

Macworld EXPO

InternetMOBILE COMPUTING

Línea de autobuses especiales desde: Aeropuerto, Plaza Castilla y Colón.

a de
drid PARQUE FERIAL JUAN CARLOS I - INFOIFEMA: TEL.: (91) 722 51 80 - 722 50 00. FAX: 722 58 07. 28042 MADRID INTERNET:http/www.simo.ifema.es

IBERIA TRANSPORTISTA OFICIAL